Optimisation des ressources de réseaux hétérogènes avec coeur de réseau MPLS

Mohamed Anouar Rachdi

■ Thèse

Préparée au Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS

En vue de l'obtention du grade de

Docteur de l'Institut National des Sciences Appliquées de Toulouse

Ecole doctorale : EDSYS

Spécialité : Systèmes Informatiques

Par **MOHAMED ANOUAR RACHDI**

Optimisation des ressources de réseaux hétérogènes avec cœur de réseau MPLS

Rapport LAAS n° : Année 2007

■ Thèse

Préparée au Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS

En vue de l'obtention du grade de

Docteur de l'Institut National des Sciences Appliquées de Toulouse

Ecole doctorale : EDSYS

Spécialité : Systèmes Informatiques

Par **MOHAMED ANOUAR RACHDI**

Optimisation des ressources de réseaux hétérogènes avec cœur de réseau MPLS

Soutenue le : 3 Mai 2007 au LAAS-CNRS à Toulouse

Président	Christian BES (UPS)
Directeur de thèse	Jean-Marie GARCIA (LAAS-CNRS)
Rapporteurs	Philippe MAHEY (ISIMA) Noëmie SIMONI (ENST)
Examinateurs	Prosper CHEMOUIL (France Telecom) Didier TRANCHIER (Adelit/INT)
Invité	Olivier BRUN (LAAS-CNRS)

Je dédie ce travail à mon père, sans qui je ne serais pas où j'en suis aujourd'hui...

Je dédie aussi ce travail à ma mère, ses prières m'ont accompagné tout le long de mes études.

REMERCIEMENTS

Les travaux présentés dans ce mémoire sont l'aboutissement de quatre années d'études effectuées au Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS (LAAS-CNRS) au sein du groupe de recherche MRS (Modélisation et contrôle des Réseaux et Signaux).

Je tiens à exprimer toute ma reconnaissance à Monsieur Malik Ghallab, directeur du laboratoire, pour m'avoir accueilli au LAAS et pour avoir mis à ma disposition tous les moyens nécessaires à la réalisation de mes travaux. Je voudrais également signaler ici l'exceptionnelle qualité des moyens informatiques et logistiques mis à disposition des chercheurs dans ce laboratoire.

J'adresse tous mes remerciements à Monsieur Jean-Marie Garcia, directeur de recherche CNRS, pour avoir dirigé cette thèse. Je le remercie plus particulièrement pour sa patience, la qualité de son encadrement et ses conseils, aussi bien scientifiques que personnels, qu'il a pu me prodiguer durant les années que j'ai passées au LAAS. Je tiens ici à lui témoigner mon amitié et mon respect.

Je remercie Monsieur Christian BES, Professeur à l'Université de Toulouse, pour avoir accepté de présider le jury de cette thèse.

Je suis très particulièrement reconnaissant envers Madame Noëmie SIMONI, professeur à l'ENST, et Monsieur Philippe Mahey, professeur à l'ISIMA, qui m'ont fait l'honneur d'être les rapporteurs de cette thèse. Je remercie plus particulièrement Monsieur Philippe Mahey pour l'ensemble de ses corrections qui ont permis de parfaire ce mémoire.

Je remercie, Monsieur Prosper CHEMOUIL, Monsieur Didier TRANCHIER et Monsieur Olivier BRUN en leurs qualités de membre du Jury.

Les remerciements exprimés ici ne seront jamais à la hauteur de son implication dans ce travail de thèse. J'exprime toute ma reconnaissance pour son aide, ses nombreux conseils, sa disponibilité, son dynamisme et sa bonne humeur à Monsieur Olivier Brun. Qu'il sois assuré, de tout mon respect et de ma profonde gratitude.

Je remercie toute l'équipe de QoS Design, pour leur rigueur et leur sérieux ainsi que pour les agréables moments passés ensemble. Une attention spéciale pour François mon « développeur Java » préféré, pour David « Monsieur C++ » ainsi que pour Cédric et Charly pour avoir osé relire ce mémoire.

Je remercie également tous les membres du LAAS, pour leur accueil, et particulièrement les nombreux amis que j'y ai rencontré.

Je ne saurais oublier de remercier toutes les personnes qui me sont chères, en particulier mes parents, pour l'aide, la confiance et le soutien dont ils ont fait preuve tout au long de ces dernières années.

Mais aussi mes deux frères, Manal pour son immense talent et pour l'énergie débordante dont il fait preuve. Il a souvent été une source de motivation supplémentaire.

Sans oublier le terrible Akram qui conjugue habilement rage de réussite avec volonté infinie de se démarquer de ses ainés. Qu'il soit rassuré ses multiples péripéties cesserons le jour ou il prendra conscience de sa propre valeur.

Enfin, j'exprime ma dernière pensée à ma petite femme Marya qui, à sa manière, m'a énormément aidé; je lui exprime mon admiration tant son courage et sa patience n'ont d'égale que son soutien indéfectible.

Mes remerciements vont aussi à tous mes amis en dehors du LAAS, auprès desquels j'ai passé d'agréables moments. J'ai beaucoup appris de la vie auprès des plus proches : AbdelMalik, Ali et sa petite famille, Mohamed, AbdelKarim et Abdessamad.

Les derniers remerciements vont, à tout le personnel technique du LAAS dont le concours a grandement facilité la réalisation de ce mémoire.

Mohamed Anouar Rachdi
Mars 2007

TABLE DES MATIERES

TABLE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION GENERALE

L'Internet d'aujourd'hui est devenu le canal de distribution privilégié de nombreuses entreprises de tout bord. Les métiers qui sont au cœur de cette révolution sont principalement ceux de l'informatique, des terminaux, de la distribution, des médias, des télécommunications, des services en général, des fournisseurs d'accès et des portails.

La vague du haut-débit va amener de nouveaux concurrents du monde de la production musicale, cinématographique et audiovisuelle. Sans oublier que le commerce par le réseau présente un gain de productivité par rapport aux canaux de distribution habituels et commence à déstabiliser la chaîne des acteurs actuels. Tout cela nous pousse à penser que le réseau à haut-débit va devenir une matière première de base de la nouvelle économie.

Ainsi les services multimédia, au premier rang desquels la télévision, ne sont plus seulement disponibles sur les réseaux hertziens ou les réseaux câblés. Ils se répandent sur la paire de cuivre grâce aux débits proposés par l'ADSL. Il est aussi prévu des programmes de télévision ou de vidéo via le paire à paire. Ces services large bande sont également accessibles sans fil, grâce aux technologies Wifi et Wimax. Ils deviennent, ou vont devenir demain, disponibles sur le mobile avec les réseaux UMTS de troisième génération. A l'inverse, on observe que les réseaux de vidéo-distribution acheminent des services de communication classiques comme la voix ou les données.

En outre, le besoin d'accéder à l'information depuis n'importe où et à n'importe quel instant est de plus en plus marqué de nos jours, et disposer en permanence d'un PC portables ou d'un téléphone est devenu quasiment vital. C'est pourquoi de nombreux opérateurs proposent déjà de prolonger certains services Internet sur les mobiles. Accéder à son mail depuis un téléphone WAP, à un bouquet de services (annuaires, jeux, localisation,…), à un répertoire de sites WAP qualifiés ou encore à un moteur de recherche, fait partie du paysage quotidien de chaque abonné.

L'enjeu pour les acteurs de l'Internet de demain sera ainsi d'enrichir la gamme des services pour s'adapter aux spécificités des terminaux, simplifier l'accès Internet et le mettre à la portée de tout un chacun, utiliser toutes les possibilités de personnalisation des services et d'augmenter « in fine » le revenu par abonné.

Avec les contenus haut-débit, les profils de trafic vont changer et le réseau doit s'y préparer. La vidéo en « streaming » est une tendance forte, au même titre que la voix sur IP. Les jeux 3D en réseau nécessitent de plus en plus de bande passante et d'interactivité. L'animation des portails large-bande peut à l'avenir se rapprocher de celle des émissions de télévision et engendrer de forts flux de trafic lors de séances de concours avec les spectateurs interactifs que seront les internautes.

Cette évolution « des mœurs » sur Internet a donc poussé au développement des réseaux en général. Il a nécessité le déploiement, l'exploitation et l'administration d'infrastructures complexes et réparties. Après le déploiement de différentes technologies d'accès radio (GSM/GPRS, UMTS, WiFi,…), on assiste maintenant à de multiples mouvements de convergence : convergence des réseaux téléphoniques et des réseaux de données, convergence des réseaux à infrastructure et des réseaux ad-hocs.

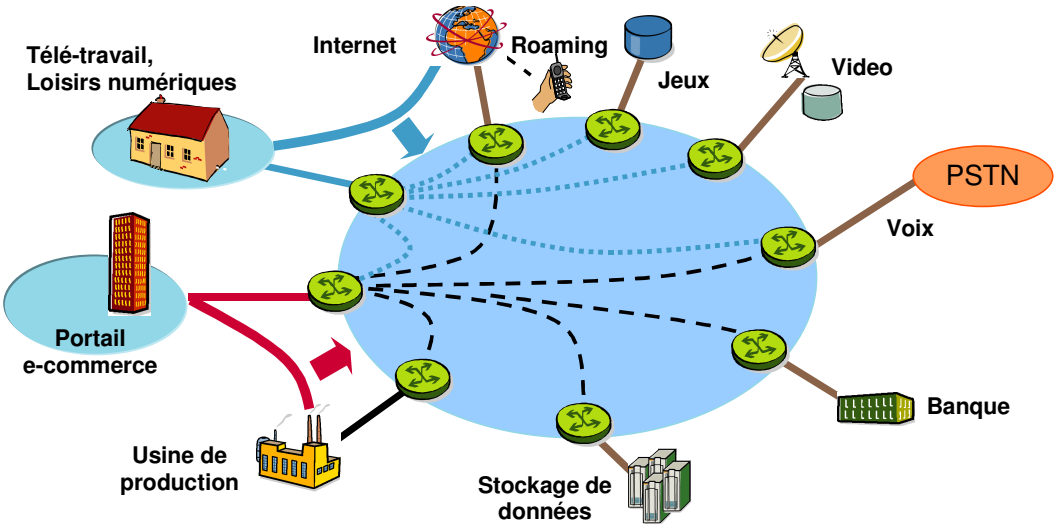


Figure 1: Convergence des services dans les réseaux du futur

Bon nombre d'applications liées aux services de la téléphonie par exemple fonctionnent dans ce contexte. Des applications, comme MSN, Skype, ou autre logiciel de Voix sur IP (VoIP), sont relativement bien connues du grand public. Cependant la qualité n'est hélas pas toujours au rendez-vous, car le réseau Internet n'offre pas encore des garanties suffisantes en termes de délais et de débits de façon générale.

Dans la mesure où les opérateurs seraient appelés à ne plus employer que de la téléphonie sur IP par exemple, il faudrait que ce service fonctionne en tous cas aussi bien que la téléphonie traditionnelle. Il s'agirait alors de pouvoir effectuer un appel d'une qualité qui ne dépende pas de la charge du réseau. La notion de qualité de service (QoS) prend tout son sens lorsqu'il y a un partage des ressources.

Il est bien entendu clair que le préjudice causé par les baisses de performances du réseau n'affecte pas toutes les applications de la même façon. Un fichier qui met une minute de plus pour être chargé n'a pas la même conséquence qu'une interruption de 60 secondes au milieu d'une conversation téléphonique.

Pour résoudre le problème posé par le partage des ressources, le réseau IP traditionnel doit connaître plusieurs évolutions.

A. Evolution des protocoles de routage

La première évolution concerne la gestion du routage. IP de part ses protocoles de routage figés tels que OSPF (Open Shortest Path First) ou encore ISIS (Intermediate System to Intermediate System), qui ne dépendent que des métriques fixes sur les interfaces du réseau, ne permet pas une gestion efficace et simple des ressources.

Il est maintenant connu que ce type de routage engendre un grand nombre de « goulot d'étranglement » dans la mesure où il ne tient pas compte de la charge du réseau et continu de router sur des chemins plus qu'engorgés.

Une première solution à ce problème est l'introduction des réseaux de type IP/MPLS (Multi-Protocol Label Switching). Cette nouvelle technologie, grâce à son routage par LSPs (Label Switched Path), permet entre autre de proposer des chemins différents que ceux proposés par les protocoles de routage de type (OSPF/ISIS).

Les LSPs peuvent être vus comme des circuits virtuels de niveau 3 qui agrègent un certain nombre de trafics. Ils sont principalement caractérisés par une source, une destination, une classe, une priorité et une bande passante.

Résoudre le problème du routage dans le contexte des réseaux IP/MPLS revient à répondre aux 3 questions suivantes :

- Quels LSPs établir dans le réseau ?
- Comment affecter les trafics aux LSPs ?
- Quels chemins utiliser pour router les LSPs ?

Les deux premières questions relèvent plus de l'ingénierie du trafic. Celles-ci sont attachées à la notion de FEC (Forwarding Equivalence Class) que l'opérateur doit établir dans son réseau. Nous avons apporté quelques éléments de réponse à ces deux questions en proposant plusieurs stratégies de création de LSPs.

La troisième question est le point central de la première partie de ce travail. En effet pouvoir imposer des routes explicites permet aussi de pré calculer les chemins des LSPs «hors ligne». Cela permet donc d'envisager l'utilisation d'algorithmes d'optimisation performants pour obtenir une meilleure solution.

Nous avons donc étudié plusieurs algorithmes de routage des LSPs. Nous proposons dans ce travail une nouvelle approche de routage des LSPs qui tient compte d'un grand nombre de contraintes opérationnelles, tels que les contraintes d'affinités ou de sécurité sur les LSPs, tout en permettant le passage à l'échelle (grands réseaux, milliers de LSPs).

B. La gestion de la QoS

La deuxième évolution concerne la gestion de la QoS. En effet les opérateurs doivent régulièrement mettre à jour leurs infrastructures pour faire face à cet accroissement du trafic sur leurs réseaux et pour assurer un niveau de QoS qui ne cesse d'augmenter avec les nouvelles applications multimédia.

Dans le contexte d'une conjecture économique difficile et très compétitive, les opérateurs ont du trouver une alternative à la solution de surdimensionnement « aveugle » des réseaux. Celle-ci passe nécessairement par le biais de l'optimisation du routage. Cette optimisation permet de mieux utiliser les ressources disponibles dans le réseau. Elle permet aussi d'éviter la dégradation des services en intégrant les indicateurs de QoS dans l'évaluation du routage.

Traditionnellement seul le critère bande passante a suscité l'intérêt des opérateurs. Seulement avec l'avènement des nouvelles applications multimédias sur les réseaux on ne peut plus se contenter de ce seul critère. Il a fallu mettre à jour les modèles en y intégrant des indicateurs de la QoS tels que le délai, la perte ou encore la gigue.

C'est pourquoi nous proposons une modélisation basée sur l'évaluation de ces indicateurs que nous intégrons dans le processus d'optimisation du routage des LSPs.

L'intégration de la gestion de la QoS dans le schéma de routage est basée sur la notion de différenciation de services. En effet DiffServ (Differentiated Services) permet de mettre en place un mécanisme de différentiation des paquets au niveau des files d'attentes. Cela

permet de traiter chaque paquet selon la classe de service à laquelle il appartient. Nous nous sommes intéressés, aux différentes politiques de « scheduling » utilisées par les opérateurs de télécommunications.

Nous proposons une modélisation très simple basée sur le modélisation par des files d'attentes M/M/1/N. Nous proposons aussi une modélisation plus complexe, mais beaucoup plus fine, basée sur le principe de la combinaison de files prioritaires et de files WFQ (Weighted Fair Queuing). Cette dernière est une émulation de l 'algorithme LLQ (Low Latency Queuing) proposé par CISCO.

C. Conception de Topologies

Durant l'étude du problème de mono-routage des LSPs nous nous sommes rendu compte que la solution dépend étroitement de la topologie du réseau. Pour ainsi dire rien ne sert d'essayer d'améliorer le routage si la topologie sous-jacente est mal conçue.

Nous avons vu précédemment que les réseaux de télécommunications sont devenus des ressources stratégiques pour les institutions tant privées que publiques et leur importance économique ne cesse d'augmenter.

L'augmentation de la connectivité des réseaux et l'intégration de plusieurs services dans un même système de communication (intégration de voix et données, téléphonie mobile, développements de la téléphonie sur plates-formes IP, etc.) a engendré une croissance significative de la complexité du métier de concepteur d'architectures de réseaux.

D'une part, sur des aspects de dimensionnement matériel puisque les structures de communication doivent fédérer un nombre croissant de points de raccordement. D'autre part, la convergence des médias où l'on cherche à faire passer sur un même support physique les données, la voix, la vidéo, entraîne l'ajout de nouveaux équipements.

Le résultat conduit en général à une augmentation de la charge du système de transmission, et pour l'architecte réseau à une préoccupation permanente d'assurer une bonne Qualité de Service (QoS : Quality Of Service) pour les utilisateurs du réseau.

Le premier réflexe de ce dernier pour répondre à cette problématique est d'utiliser des matériels de communication puissants avec beaucoup de mémoire, des processeurs rapides et des lignes à hauts débits. Partant de cet état de fait, l'élaboration d'une topologie optimale nous est apparue, dans ce contexte, un point important à étudier.

Le principal problème étudié consiste à minimiser le coût total du système de communication tout en garantissant sa résilience.

Pour ce faire nous proposons une modélisation qui tient compte du trafic et des coûts d'équipements. Nous élaborons deux approches la première basée sur la programmation linéaire en nombres entiers. La deuxième approche est une heuristique qui combine une technique de « clustering » et une technique de recherche locale.

D. Plan de la thèse

Le premier chapitre de la thèse présente le contexte des réseaux IP/MPLS et montre la complexité du routage dans de tels réseaux. Nous présentons le routage classique dans les réseaux IP ainsi que le routage par LSPs dans les réseaux MPLS. Nous présentons à la fin de ce chapitre une première formulation du problème de mono-routage des LSPs. Cette formulation permet d'optimiser l'allocation de ressources et d'optimiser la bande passante résiduelle du réseau.

Le deuxième chapitre est dédié à la gestion de QoS dans les réseaux MPLS. Les mécanismes utilisés pour mettre en œuvre cette QoS (IntServ / DiffServ) sont présentés. Nous en discutons les avantages et les inconvénients de ces mécanismes. Nous proposons suite à cela deux modélisations analytiques de la QoS. La première est basée sur les files M/M/1/N et la deuxième est une émulation du protocole LLQ proposé par CISCO. Nous proposons ensuite une extension du problème de mono-routage qui intègre les contraintes de QoS au niveau des LSPs.

Le troisième chapitre traite des méthodes d'optimisation non-linéaires. Nous considérons plusieurs de ces méthodes, parmi les plus connues dans la littérature, afin de déterminer la plus efficace pour notre problème. Le but étant d'approcher la solution du problème de mono-routage à l'aide de la solution multi-routée. Nous présentons dans ce chapitre une nouvelle méthode ILSP (Iterative Loading Shortest Path) pour la génération de chemins candidats. Nous terminons ce chapitre par un comparatif des performances des méthodes étudiés sur des topologies tests.

Le quatrième chapitre traite du problème du mono-routage des LSPs. Nous présentons l'algorithme CISCO Pcalc. Nous présentons par la suite un nouvel algorithme innovant ILSP-OLS-ACO qui combine la qualité des chemins candidats générés par ILSP, la précision de la solution multi-routées par OLS et la qualité de la solution mono-routée par ACO.

Le cinquième chapitre traite de la sécurisation des réseaux IP/MPLS. Nous présentons les différentes techniques de sécurisation disponibles (Backup, Fast Reroute, Multilayer). Nous présentons suite à cela les différentes extensions de l'algorithme ILSP afin de générer des chemins candidats en tenant compte de la contrainte de sécurisation. Nous présentons aussi les algorithmes de calcul des Backup est des Fast Reroute avec ou sans notion de SRLG.

Le sixième chapitre traite de la conception de topologie d'accès. Nous présentons une modélisation du problème qui tient compte du trafic ainsi que des coûts d'équipements. Nous proposons une version décomposée du problème initial basée sur le calcul des configurations « matérielles » optimales au niveau des sites. Nous proposons pour cette modélisation une approche exacte (Branch and Bound amélioré) et une approche heuristique (Lagrangien et formulation duale). Nous proposons aussi une heuristique qui exploite une technique de « clustering » ainsi qu'un algorithme de recherche locale. Nous présentons ensuite les résultats obtenus par les différentes méthodes sur des topologies tests.

« Je pense qu'il y'a un marché pour peut-être 5 ordinateurs dans le monde »

Thomas Watson, président d'IBM,
1943.

Chapitre 1 - FORMULATION DU PROBLEME DE MONO-ROUTAGE DES LSPS DANS LE CONTEXTE DES RESEAUX IP/MPLS

A. Introduction

Un réseau peut être vu comme un ensemble de ressources mises en place pour offrir un ensemble de services. C'est l'évolution des services et des trafics qui en découlent qui a piloté, dans les dernières années, l'évolution technologique qui a permis d'augmenter la capacité et les fonctionnalités des réseaux. Ainsi, par exemple, le succès des services de l'Internet a engendré une explosion de trafics qui a mené les opérateurs à utiliser de nouvelles technologies dans le cœur de réseau tel que l'IP (Internet Protocol) sur ATM (Asynchronous Transfer Mode), le PoS (Packet over Sonet/SDH) , ou encore MPLS (Multi Protocol Label Switching).

Dans ce qui suit nous allons nous intéresser à IP et MPLS. Nous commencerons par décrire les réseaux traditionnels fédérés par IP. Nous introduirons par la suite le protocole MPLS et ce qu'il apporte comme avantage par rapport à IP. Nous allons aussi souligner les interactions entre ces deux protocoles ainsi que leur mode de cohabitation.

Comme nous l'avons présenté la gestion des ressources d'un réseau est devenue un point crucial dans le processus de planification. A la lumière des nouvelles possibilités offertes par la technologie MPLS nous abordons dans ce chapitre ce problème. Nous proposons une première formulation du problème de mono-routage des LSPs dans les réseaux MPLS. Cette formulation est basée sur l'approche bande passante.

B. Quelques éléments sur IP

IP est un protocole qui permet l'adressage des machines et le routage des paquets de données [RFC791]. Il correspond à la couche 3 (réseau) de la hiérarchie des couches ISO. Son rôle est d'établir des communications sans connexion de bout en bout entre des réseaux, de délivrer des trames de données (datagram) « au mieux (best effort) », et de réaliser la fragmentation et le ré-assemblage des trames pour supporter des liaisons n'ayant pas la même MTU (Maximum Transmission Unit, c'est-à-dire la taille maximum d'un paquet de donnée).

B.1. L'adressage IP

Un des éléments essentiels d'IP est la couverture mondiale qu'assure le protocole. Ceci est fait grâce à une gestion d'adresses uniques. L'adressage IP permet d'identifier de façon unique une interface dans une machine connectée à un réseau. Une machine ayant plusieurs interfaces est communément appelée un routeur. Elle est capable de recevoir et de renvoyer des paquets de données par le biais de différentes interfaces.

Elle utilise pour cela une table de routage qui, dans sa version la plus simple, contient un certain nombre d'adresses internet complètes ou sous forme de masques (liste non exhaustive), l'interface à utiliser pour atteindre chacune d'elles, et une ou plusieurs interfaces par défaut vers lesquelles envoyer les paquets d'adresse inconnue. Un paquet IP est composé d'une en-tête et de données. La zone Data (Données) contient les données utiles provenant des applications (couches supérieures) à transférer via le réseau.

Les informations principales contenues dans l'entête d'un paquet IP sont : les adresses de la source et de la destination, le type de service (ToS), et la longueur totale du paquet.

B.2. Le routage dans les réseaux IP

Le routage IP est un routage de proche en proche. Dans un routeur, une table de routage IP contient la liste des adresses connues localement afin d'associer une interface de sortie à toutes les destinations (next-hop). Lorsqu'un paquet de destination inconnue (non locale) est reçu, le paquet est envoyé vers le routeur de frontière par défaut.

Le routage se compose de deux éléments essentiels :

- Détermination des chemins optimaux de routage,
- Envoi des paquets IP à travers des réseaux.

L'envoi des paquets est somme toute une tâche assez simple. La détermination d'un routage optimal est par contre beaucoup plus complexe.

Les routeurs dans l'Internet sont organisés de manière hiérarchique. Des routeurs sont utilisés pour envoyer les informations dans des groupes de routeurs particuliers sous la même responsabilité administrative. Ces groupes sont appelés des « Autonomous Systems » (AS). Ces routeurs, appelés Interior Gateway Router, utilisent des protocoles de routage appelés Interior Gateway Protocols (IGP). D'autres routeurs permettent à différents Autonomous Systems de communiquer entre eux. Ils utilisent des protocoles de routage appelés Exterior Gateway Protocols (EGP).

En échangeant régulièrement des informations, les routeurs construisent une image de la topologie du réseau ce qui permet de créer les tables de routage. Les tables de routage contiennent en tout nœud, le prochain routeur à utiliser (next hop) pour une destination donnée. Le routage IP spécifie donc **bond par bond** (**hop by hop)** comment envoyer des paquets pour une adresse destination donnée.

En fait, la route complète de l'origine jusqu'à la destination n'est pas connue pendant le transit du paquet. Seuls les nœuds intermédiaires sont déterminés un par un parmi les voisins d'un nœud donné. IP opère donc en mode non connecté.

Les principaux objectifs à atteindre par un protocole de routage sont :

- ***Optimisation*** (sélectionner le meilleur chemin selon les critères choisis),
- ***Simplicité*** (le plus simple possible),
- ***Robustesse et souplesse*** (faire face à toutes sortes d'imprévus),
- ***Convergence rapide*** (recalculer les routes rapidement pour éviter les boucles, les pannes réseau,…).

Pour la suite, nous allons nous intéresser aux deux protocoles de routage les plus utilisés dans l'Internet : OSPF (Open Shortest Path First) et BGP (Border Gateway Protocol).

B.2.1. Le protocole OSPF

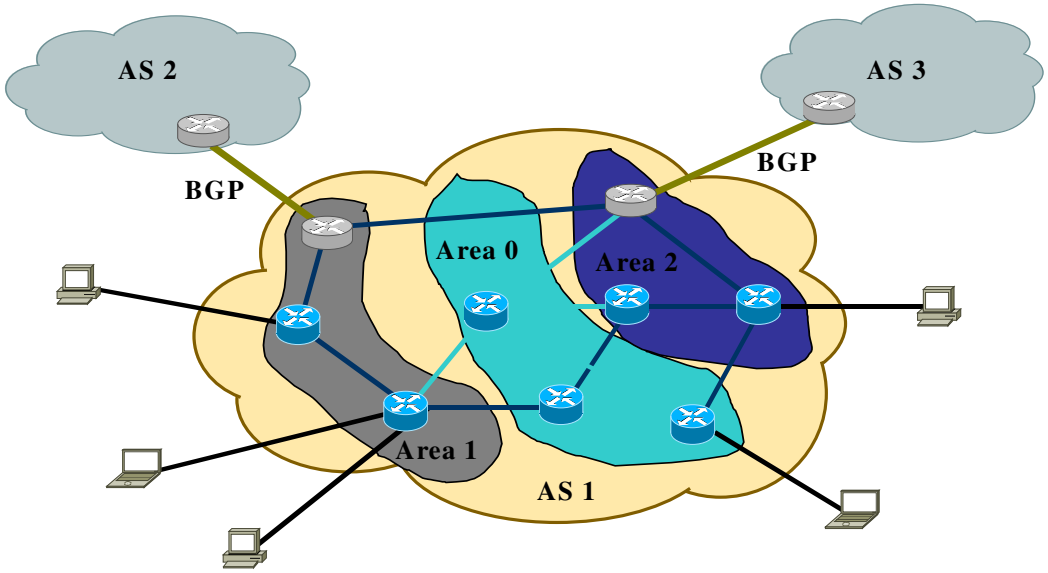


Figure 2: Exemple d'AS composé de trois zones

Le protocole OSPF (Open Shortest Path First) est un protocole de routage à état de lien (Link-State) créé par l'IETF [RFC2328]. Il est actuellement l'IGP (Interior Gateway Protocol) le plus répandu.

L'état de lien permet au routeur d'avoir une vision globale du réseau et de sa topologie et l'émission des mises à jour n'est déclenchée qu'en cas de modifications topologiques.

Pour générer la table de routage, le protocole OSPF utilise un arbre du plus court chemin d'abord (SPF Tree : Short Path First Tree) et un algorithme du plus court chemin d'abord (SPF Algorithm) appelée aussi algorithme de Dijkstra, qui détermine le meilleur chemin en terme de coût.

Le coût du chemin dépend de métriques figées par l'opérateur sur chacune des interfaces du réseau. La métrique, est un coût sans dimension aucune valeur n'est imposée. Libre à l'opérateur de choisir la valeur qu'il juge pertinente (CISCO préconise la valeur 10^8/débit).

Le protocole OSPF est capable de différencier les classes de service, grâce au champ ToS, permettant ainsi un routage différent selon le type de trafic dans le

réseau. Les tables de routages pouvant être différentes selon les classes de services, on peut utiliser une métrique différente pour chacune d'elle.

Lorsque plusieurs routes sont équivalentes d'un point de vue métrique (même coût), le partage de charge (Load-Balancing) entre ces routes peut être employé. Le nombre de routes sur lesquelles on peut faire du partage de charge est en général limité (à quatre sur certains routeurs).

Un réseau OSPF peut-être composé d'un ou plusieurs domaines de routage, les aires (Areas), au sein d'un même système autonome : l'aire 0 constitue l'aire de backbone.

B.2.2. Le protocole BGP

Le protocole **BGP** (**B**order **G**ateway **P**rotocol) est un protocole de routage entre AS [RFC1771]. BGP est utilisé pour l'échange d'informations de routage entre AS de l'Internet. Les informations de routage se présentent sous la forme de la suite des numéros d'AS à traverser pour atteindre la destination. C'est BGP qui assure la propagation des routes à l'échelle mondiale.

Quand BGP est utilisé entre AS, le protocole est connu sous le nom de BGP Externe (e-BGP). Si BGP est utilisé à l'intérieur d'un AS pour échanger des routes, alors le protocole est connu sous le nom de BGP interne (i-BGP). Les routeurs de frontière communiquant par i-BGP doivent être complètement maillés (mais pas forcément directement connectés). Avec BGP4, on peut toutefois utiliser les techniques dites de route reflector et de confédérations pour diminuer le nombre d'échanges de messages à l'intérieur d'un grand AS.

BGP est un protocole très robuste et très « scalable » : les tables de routage BGP de l'Internet comprennent plus de 90000 routes. Si BGP a atteint ce niveau de « scalabilité », c'est essentiellement parce qu'il associe aux routes des attributs permettant de définir des politiques de routage entre AS (c'est aussi grâce au sous-adressage).

Les voisins BGP échangent toutes leurs informations de routage au début, lors de l'établissement d'une connexion TCP entre eux. Par la suite, les routeurs BGP n'envoient à leur voisins que des mises à jour des tables de routage lorsqu'un changement de route est détecté ou qu'une nouvelle route est apprise. Ainsi, les routeurs BGP n'envoient pas des mises à jour périodiquement et ne diffusent que le chemin « optimal » vers un réseau destination.

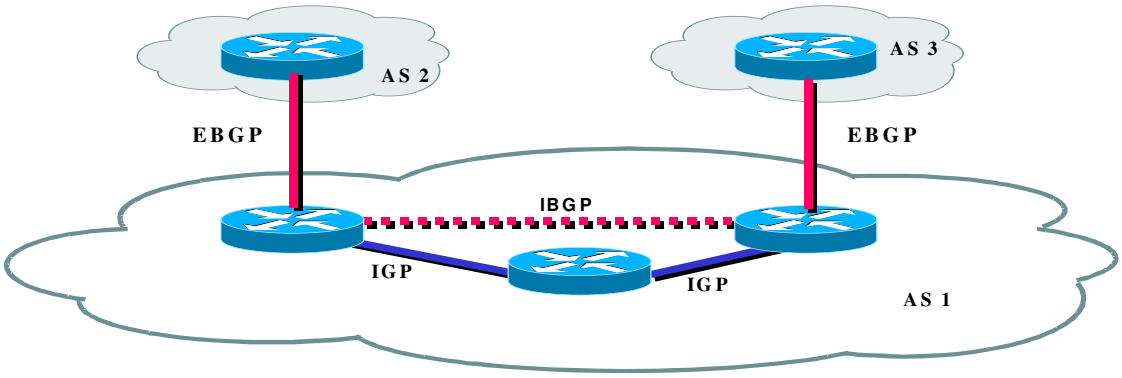


Figure 3: Routeurs BGP

C. Le concept MPLS

Avec l'augmentation du débit des liens de communication, les routeurs de l'Internet menaçaient de devenir des goulots d'étranglement. L'IETF a ainsi défini le nouveau protocole MPLS [RFC3031], [RFC3032] avec deux objectifs principaux :

- Permettre un acheminement rapide des paquets IP en remplaçant la fonction de routage par une fonction de commutation qui est beaucoup plus rapide, la taille des matrices de commutation étant très petite. La recherche se fait suivant le principe « exact match » au lieu de « best match ».
- Faciliter l'ingénierie réseau en fournissant aux opérateurs la maîtrise de l'acheminement des données, qui étaient très complexes avec des protocoles comme OSPF.

Le protocole MPLS, basé sur le paradigme de changement de label, dérive directement de l'expérience acquise avec l'ATM (étiquettes VPI/VCI). Ce mécanisme est aussi similaire à celui de Frame Relay ou de liaisons PPP. L'idée de MPLS est de rajouter un label de couche 2 aux paquets IP dans les routeurs frontières d'entrée du réseau. Le réseau MPLS va pouvoir créer des **LSPs** (Label Switching Path), similaires aux PVC d'ATM.

Ces LSPs définissent une route de bout en bout par une concaténation de labels. A l'entrée du réseau MPLS, le label est rajouté à l'entête IP par le routeur frontière **LER** (Label Edge Routers) puis à la sortie du réseau MPLS le label est retiré par un autre routeur frontière, ce qui permet de retrouver le paquet IP original. Les routeurs du cœur de réseau, appelés **LSR** (Label Switching Routers), vont router les paquets de proche en proche par commutation de labels, « oubliant » ainsi les adresses IP.

L'entête MPLS se situe entre les entêtes des couches 2 et 3, où l'entête de la couche 2 est celle du protocole de liaison et celle de la couche 3 est l'entête IP. L'entête est composée de quatre champs:

- Le champ **Label** (20 bits),
- Le champ **Exp** ou **CoS** (3 bits) pour la classe de service (Class of Service),
- Un bit **Stack** pour supporter un label hiérarchique (empilement de labels),
- Et un champ **TTL** (Time To Live) pour limiter la durée de vie du paquet (8 bits). Ce champ TTL est le même que pour IP.

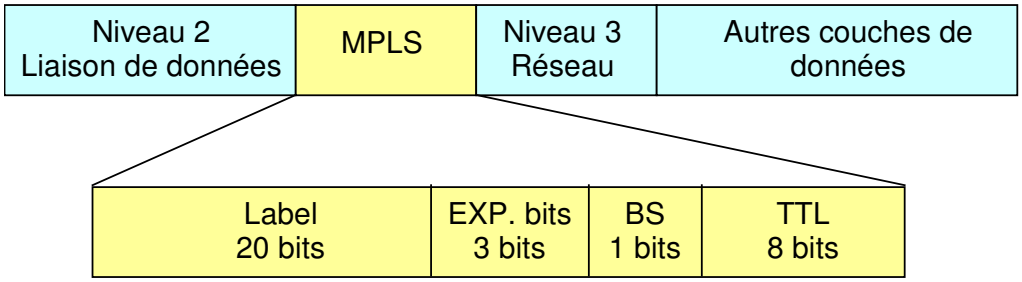


Figure 4: Entête MPLS

Remarque : dans les réseaux ATM et Frame Relay, cette entête n'est pas utilisée. Ce sont les champs VPI/VCI dans ATM et DLCI dans Frame Relay qui servent de label.

Les grandes innovations par rapport au routage IP traditionnel (IGP, EGP) sont :

- La manipulation de tables de routage de bien plus petite taille et des champs de petite taille pour les labels conduisant ainsi à des temps de commutation extrêmement rapides,
- La détermination de routes (LSPs) par classe de service et pas seulement par adresse de destination,
- un ensemble de règles et de paramètres permettant de calculer un routage « optimal » (autre que le plus court chemin) pour l'ingénierie du trafic et la qualité de service,
- Des mécanismes de reroutage en cas de panne avec la possibilité de réserver des LSPs de secours pour certains trafics.

MPLS est aussi particulièrement adapté à un service différencié de type DiffServ. En effet, dans son concept, on retrouve la notion de routeurs aux frontières et de routeurs de cœur de réseau (edge/core router) ainsi que la notion d'agrégation de flots.

Le principe est simple : il s'agit d'étiqueter les paquets IP (labélisation) suivant le chemin qu'ils doivent suivre dans le domaine MPLS et suivant leur classe de service. On appelle **FEC** (Forwarding Equivalent Class) les classes d'équivalence de transfert. Les paquets d'une même FEC empruntent le même chemin et subissent les mêmes traitements dans le réseau.

L'intégration du modèle DiffServ dans un réseau MPLS [RFC3812] est très simple puisqu'il suffit d'affecter un label à un paquet en fonction de sa classe de service et de sa destination (FEC). A l'heure actuelle, les projets de réseaux IP multiservices concernent d'ailleurs principalement des réseaux DiffServ MPLS.

L'un des objectifs initiaux de MPLS était d'accroître la vitesse du traitement des datagrammes dans l'ensemble des équipements intermédiaires. Cette volonté, avec l'introduction des gigarouteurs, est désormais passée au second plan. Depuis, l'aspect « fonctionnalité » a largement pris le dessus sur l'aspect « performance », avec notamment les motivations suivantes :

- Intégration IP/ATM
- Création de VPN
- Flexibilité : possibilité d'utiliser plusieurs types de media (ATM, FR, Ethernet, PPP, SDH).
- Differentiated Services (DiffServ)
- Routage multicast
- MPLS peut assurer une transition facile vers l'Internet optique. MPLS n'étant pas lié à une technique de niveau 2 particulière, il peut être déployé sur des infrastructures hétérogènes (Ethernet, ATM, SDH, WDM, etc.).
- MPLS permet la prise en charge de la qualité de service (DiffServ, Cisco Guaranteed Bandwidth).
- Traffic Engineering permettant de définir des chemins de routage explicites dans les réseaux IP (avec RSVP ou CR-LDP). Grâce à cette fonctionnalité, MPLS permet une simplification radicale des réseaux.

Les labels peuvent être associés à un chemin, une destination, une source, une application, un critère de qualité de service, etc. ou une combinaison de ces différents éléments. Autrement dit, le routage IP est considérablement enrichi sans pour autant voir ses performances dégradées (à partir du moment ou un datagrame est encapsulé, il est acheminé en utilisant les mécanismes de commutation de niveau 2).

Un des services les plus importants que permet MPLS est la possibilité de créer des réseaux privés virtuels (VPN) de niveau 3. Ainsi, des services de voix sur IP, de multicast ou d'hébergement de serveurs web coexistent sur une même infrastructure. La modularité de MPLS et la granularité des labels permettent tous les niveaux d'abstraction envisageables.

Enfin, l'une des forces de MPLS est sa capacité d'agrégation de flux : la possibilité de réunir le trafic entrant dans un routeur via plusieurs LSPs dans un seul et unique LSP sortant. Une telle configuration correspond à monter une connexion multi-point à point (fonction de VC merge en ATM), comparable à un arbre inversé dont le routeur de sortie serait la racine. Cette fonction permet de réduire au maximum le nombre de connexions que les routeurs de cœur de réseau ont à gérer.

Dans ce même but, MPLS introduit un concept de hiérarchie au niveau des LSP et d'empilement des labels. Il est donc possible de construire des LSP, encapsulés dans un autre LSP, lui-même encapsulé dans un LSP, etc. Ce concept d'encapsulation rappelle évidemment la possibilité en ATM de mettre des VC dans des VP. Cependant, en MPLS, le nombre de niveaux d'encapsulation (ou de hiérarchie) n'est pas limité à 2.

C.1. Distribution des labels

Les LSR se basent sur l'information de label pour commuter les paquets au travers du cœur de réseau MPLS. Chaque routeur, lorsqu'il reçoit un paquet taggué, utilise le label pour déterminer l'interface et le label de sortie. Il est donc nécessaire de propager les informations sur ces labels à tous les LSR. Pour cela, suivant le type d'architecture utilisée, différents protocoles sont employés pour l'échange de labels entre LSR ; en voici quelques exemples :

- TDP/LDP (Tag/Label Distribution Protocol) : mapping des adresses IP unicast.
- CR-LDP, RSVP-TE : utilisés en Traffic Engineering pour établir des LSP en fonction de critères de ressources et d'utilisation des liens.
- MP-BGP (MultiProtocol Border Gateway Protocol) pour l'échange de routes VPN.

Chaque paquet MPLS est susceptible de transporter plusieurs labels, formant ainsi une pile de labels, qui sont empilés et dépilés par les LSR. Cela permet entre autre d'interconnecter plusieurs réseaux, chacun possédant son propre mécanisme de distribution des labels.

Lorsqu'un LSR commute un paquet, seul le premier label est traité. Cette possibilité d'empiler des labels, désignée sous le terme de Label Stacking, est aussi utilisée par le Traffic Engineering et MPLS / VPN.

C.1.1. Le protocole CR-LDP

CR-LDP [RFC3988] est une version étendue de LDP, où CR correspond à la notion de « routage basé sur les contraintes des LSP ». Tout comme LDP, CR-LDP utilise des sessions TCP entre les LSR, au cours desquelles il envoie les messages de distribution des étiquettes. Ceci permet en particulier à CR-LDP d'assurer une distribution fiable des messages de contrôle.

Les échanges d'informations nécessaires à l'établissement des LSP utilisant CR-LDP sont décrits dans la figure suivante :

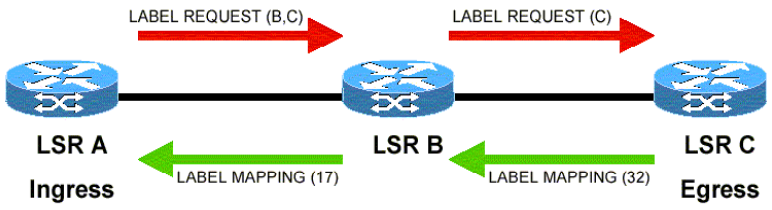


Figure 5: Etablissement d'un LSP par CR-LDP

C.1.2. Le protocole RSVP – TE

Le protocole RSVP utilisait initialement un échange de messages pour réserver les ressources des flux IP à travers un réseau. Une version étendue de ce protocole RSVP-TE [RFC3209], en particulier pour permettre les tunnels de LSP, autorise actuellement RSVP à être utilisé pour distribuer des étiquettes MPLS.

RSVP est un protocole complètement séparé de la couche IP, qui utilise des datagrammes IP ou UDP (User Datagram Protocol) pour communiquer entre LSR. RSVP ne requiert pas la maintenance nécessaire aux connexions TCP, mais doit néanmoins être capable de faire face à la perte de messages de contrôle.

Les échanges d'informations nécessaires à l'établissement de LSP permettant les tunnels de LSP et utilisant RSVP sont décrits dans la figure suivante :

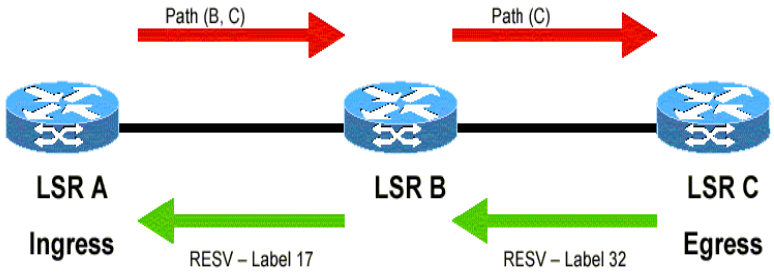


Figure 6: Etablissement LSP par RSVP-TE

C.2. Définition des LSPs suivant les RFC de l'IETF

Le LSP est une succession de routeurs LER et LSR, c'est le chemin que vont suivre les paquets dans le réseau MPLS. Un LSP est défini par les attributs suivants :

- Le LSR d'entrée (Ingress LSR) et de sortie (Egress LSR),
- La (ou les) FEC qu'il transporte,
- La bande passante réservée,
- Mandatory : est activé si le chemin est imposé au LSP,
- Hierarchy : peut être activé lorsque plusieurs chemins sont disponibles,
- Affinity : (ou couleur) permet de marquer les différentes ressources du réseau. On peut alors imposer à un LSP d'utiliser certaines classes de ressources (et donc pas d'autres). Par exemple n'utiliser que des liaisons chiffrées, cryptées ou satellites etc,
- Setup Priority : permet d'affecter une priorité aux LSPs. Cette priorité est tout d'abord utilisée pour déterminer l'ordre dans lequel les LSPs doivent être établis, et donc l'ordre dans lequel ils vont accéder aux ressources du réseau,
- Holding Priority : permet à un LSP de remplacer un autre LSP de son chemin, si cela est nécessaire. Cela est utile si des LSPs de haute priorité doivent être absolument reroutés en cas de panne et remplacer d'autres LSPs moins prioritaires. Cette situation peut arriver en cas de manque de bande passante,
- Fast ReRouting : indique la présence d'un chemin de contournement à utiliser en cas de pannes,

- Explicite Routing : permet de spécifier entièrement (strict) ou partiellement (loose) le chemin du LSP,
- Adaptivity : permet de spécifier si un LSP est réoptimisable (c'est-à-dire si on lui cherche un autre chemin de secours) en cas de problème (panne détectée d'un élément),
- Policing : spécifie les opérations à effectuer en cas de trafics non cohérents avec les contrats,
- AutoBandwidth : permet au LSP d'adapter dynamiquement la bande passante réservée par rapport au trafic présent. On peut spécifier la fréquence avec laquelle on demande la bande passante, spécifier le maximum de bande passante sur le tunnel et la bande passante minimale.

A ces attributs, on peut ajouter les attributs relatifs à l'évaluation de performances tel que le délai, la perte et la gigue maximum tolérés (par classes de services). Plusieurs informations nécessaires au routage des LSPs sont distribuées périodiquement :

- Lorsqu'un changement de la Bande Passante est détecté,
- Lorsqu'une modification de la configuration des liens est détectée,
- Etc.

Ensuite, le calcul du chemin (Path Computation) répondant aux différents critères et contraintes est réalisé. Cette détermination a lieu avant la création réelle (physique) du LSP. Ce calcul de chemin est effectué lors de la demande de création d'un nouveau LSP, lors d'une panne, ou lors d'une demande de réoptimisation.

D. Rôle du « Traffic Engineering » dans MPLS

Par TE-MPLS « Traffic Engineering MPLS », il faut comprendre, établissement de connexions « à la demande », ingénierie de trafic, gestion des routes, gestion des ressources, gestion de l'écoulement des flux de trafic à travers un réseau IP.

Le TE-MPLS introduit, la possibilité de faire suivre à des paquets IP un chemin à travers le réseau qui ne correspond pas forcément au plus court chemin de niveau 3 (issu du protocole de routage interne du réseau, i.e. RIP, OSPF, IS-IS, EIGRP, etc.), ceci afin de mieux en gérer les ressources.

MPLS et ses LSPs constituent dès lors, grâce au TE-MPLS, un outil de gestion et d'optimisation de l'utilisation des ressources d'un réseau IP : les LSP sont placés par l'opérateur selon un chemin que celui-ci peut avoir préalablement choisi et déterminé.

Le TE-MPLS, permet donc de router les LSPs à travers le réseau en tenant compte de la disponibilité des ressources et du trafic courant et attendu. La classe de service et la qualité de service requises pour les données peuvent aussi être prises en compte.

Le TE-MLPS peut être réalisé manuellement ou suivant un processus automatisé, réagissant aux informations relatives à l'état du réseau (charges des liens, pannes, nouveaux trafics).

Grâce à son TE, MPLS permet de gérer des services nécessitant différents niveaux de priorités, tel le transport de voix ou de vidéo sur Internet, sans avoir recours à une politique de surdimensionnement des réseaux physiques.

Le TE-MPLS est donc une seconde valeur ajoutée à la solution MPLS. Il permet en effet à l'opérateur une gestion plus efficace de son réseau grâce à la possibilité de tailler des « LSP » en fonction des classes de services (CoS) et des FEC (Forwarding Equivalent Class).

Cela est en plus combiné à un système de priorités servant à favoriser l'accès aux ressources pour certains trafics. Cette « intelligence » offre au client final un réseau répondant à tous ses critères de QoS, une gestion et une optimisation des ressources pour l'opérateur.

D.1. Comment constituer une FEC ?

C'est une fonctionnalité qui permet de spécifier quels paquets vont être aiguillés dans un LSP. La FEC identifie l'ensemble des paquets IP qui seront envoyés sur un LSP. Une FEC est constituée d'un ensemble d'éléments. Chaque élément identifie des paquets qui peuvent utiliser ce LSP.

Les éléments de FEC sont des types suivants (la liste n'est pas exhaustive et on peut donc en avoir autant que désiré) :

- Famille d'adresses,
- Préfixe d'adresse (peut aller jusqu'à l'adresse complète),
- Une adresse complète de machine,
- Etc.

Entre deux LER MPLS, on peut donc regrouper tous les trafics dans une même FEC pour les envoyer dans le même LSP ou séparer certains de ces trafics dans des FEC différentes afin de les placer sur des LSPs différents. Cette dernière décomposition permet d'avoir une granularité très fine qui conduit à une meilleure répartition des trafics dans le cœur de réseau par autant de LSP que nécessaires. Les cas suivants peuvent se présenter :

- Un paquet peut être envoyé sur un LSP dont un élément de FEC indique le routeur Egress si et seulement si aucun autre LSP n'a comme FEC l'adresse de destination du paquet. L'adresse de destination du paquet est donc toujours prioritaire.

- Un paquet peut correspondre à deux FEC de deux LSPs distincts. Le premier contient une FEC ayant même préfixe d'adresse, et le second a une FEC donnant l'adresse complète. Dans ce cas c'est le LSP avec la FEC donnant l'adresse complète qui est choisi.

- Un paquet qui n'a pas la bonne adresse de machine destination (dans le cas où le LSP a une FEC qui positionne une adresse machine), ne peut être envoyé sur ce LSP même si cette adresse identifie un routeur Egress. C'est

un cas important à noter car il signifie qu'on ne peut forcer un paquet à sortir par un autre Routeur Egress que celui prévu à priori.

En conséquence, un Ingress LER peut affecter un même label MPLS à des flots n'ayant pas la même destination comme représenté sur la figure ci-dessous.

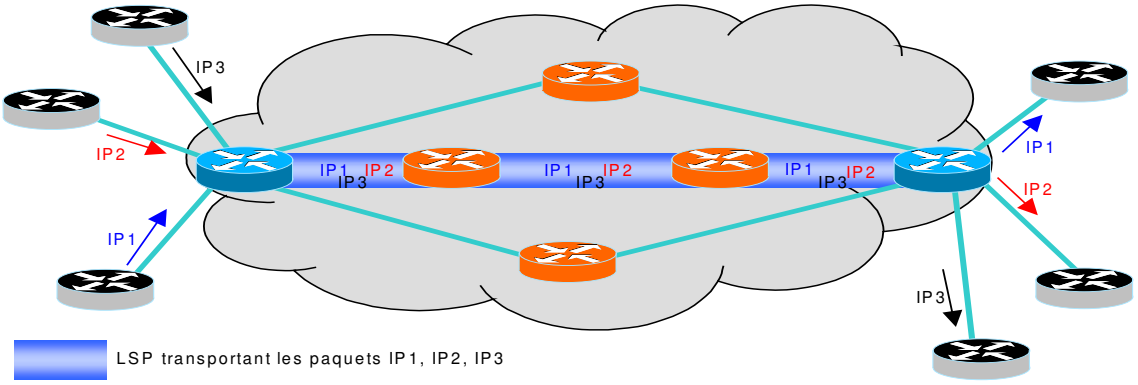


Figure 7: Scénario 1 : Un LSP pour toutes les classes

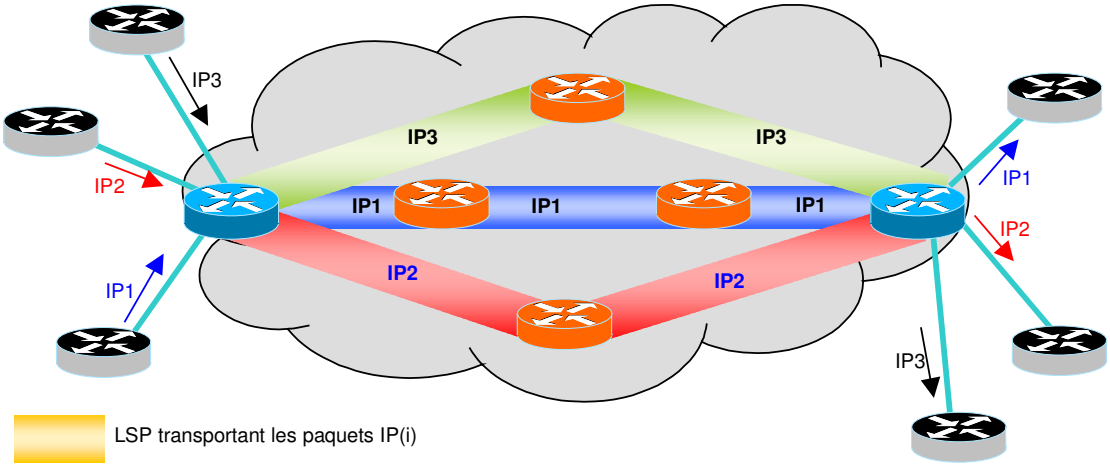


Figure 8: Scénario 2 : un LSP par destination et par classe

Les possibilités offertes sont diverses, c'est à l'opérateur de définir son niveau de granularité. Ainsi plus la granularité est grande plus il pourra déployer des services spécifiques à chaque profil client mais plus le nombre de LSPs à gérer dans son réseaux sera contraignant. A l'inverse moins la granularité sera forte moins il aura de LSPs à gérer, par contre il ne pourra plus différencier ses services à souhait.

Le TE-MPLS est un outil qui permet à l'opérateur de trouver le juste milieu entre ces deux extrêmes afin de mieux gérer son réseaux.

E. MPLS et le Multicast

Avec la croissance des services utilisant la diffusion de données telles que la vidéo, la visioconférence, ou encore la voix sur IP, la consommation en bande passante est devenue un point cruciale de l'ingénierie de réseaux. Le service multicast est apparu comme la solution naturelle et adéquate pour garantir la QoS fournie aux utilisateurs grâce à la bande passante supplémentaire disponible sur le réseau.

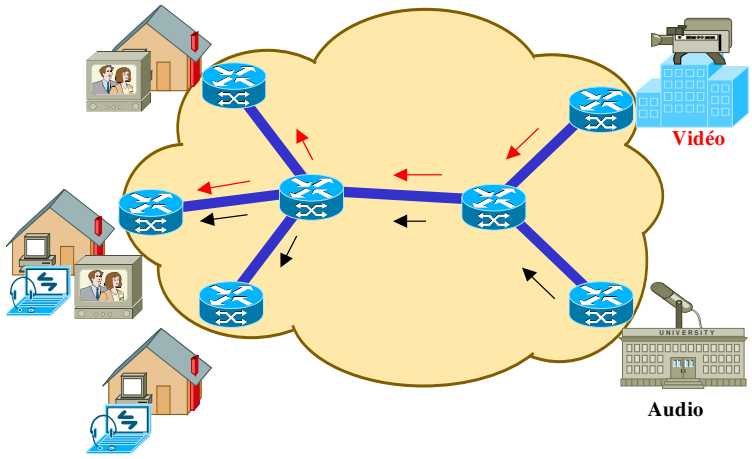


Figure 9: Multicast dans les réseaux MPLS

Comme présenté précédemment MPLS grâce à son Traffic Engineering permet une gestion efficace de la bande passante du réseau. Il a fallu adapter le service multicast à MPLS [RFC3353] pour tirer profit de la cohabitation de ces deux technologies complémentaires. En effet les arbres multicast utilisés dans des réseaux MPLS permettent de minimiser l'utilisation de la bande passante et MPLS avec son Traffic Engineering permet de redistribuer celle-ci de façon appropriée aux autres utilisateurs.

MPLS peut être employé dans un réseau pour expédier le trafic unicast à l'aide de chemins explicites. De même MPLS peut être employé pour le trafic multicast en utilisant des arbres explicites. Mais le trafic multicast possède des caractéristiques spécifiques dues à la nature du routage multicast [OOM02]. En effet, le routage multicast d'Internet se base sur l'adresse IP multicast et c'est pourquoi il est très difficile d'agréger le trafic multicast puisque les destinataires appartenant au même groupe multicast peuvent être extrêmement dispersés. De plus, la structure arborescente du multicast pourrait nécessiter d'établir des LSP point à multipoint ou même des LSP multipoint à multipoint. Dans les implémentations actuelles de MPLS, les LSP point à point sont privilégiés même si MPLS n'exclut pas d'autres types de LSP.

Afin de construire les arbres multicast plusieurs protocoles ont été proposés. Nous citons parmi ceux ci :

PIM-MPLS [FAR00] qui utilise les messages d'adhésion du protocole multicast PIM-SM [EST98] pour distribuer des labels MPLS et construire ainsi l'arbre multicast MPLS (arbre multicast dont le routage est effectué au niveau de la couche "Liaison de données"). Les messages d'adhésion de PIM-SM sont modifiés pour porter un label MPLS alloué par un LSR en aval.

Aggregated Multicast [CUI03] suppose que plusieurs groupes multicast peuvent partager un même arbre multicast au lieu de construire un arbre multicast pour chaque groupe multicast. Ceci réduit le nombre d'états de routage dans les routeurs du domaine et, également, minimise la gestion de l'arbre. Chaque groupe multicast est associé à un arbre agrégé. Pour la gestion de l'arbre agrégé et l'association entre les groupes multicast et les arbres agrégés, une entité de gestion appelé tree manager est introduite.

MMT [DRAFT05] (MPLS multicast tree) construit un arbre multicast dans un réseau MPLS en considérant seulement les routeurs de branchement de cet arbre. En limitant la présence d'états de routage multicast aux routeurs de branchement, le protocole MMT convertit les flux multicast en multiple flux quasi-unicast. Dans MMT, au lieu de construire un arbre pour chaque canal multicast individuel dans le réseau cœur, on peut avoir plusieurs canaux multicast qui partagent des branches de leurs arbres. Les LSP unicast sont utilisés entre les routeurs de branchement de l'arbre multicast. Cette méthode, permet de réduire la quantité d'informations à mémoriser dans les routeurs et assure la résistance au facteur d'échelle.

Pour traiter le multicast nous avons retenu la solution des LSPs point à point pour la construction d'arbre multicast basée sur le protocole MMT. Nous proposons en outre, deux stratégies d'agrégation pour les arbres multicast.

Celles-ci sont illustrées dans la figure qui suit.

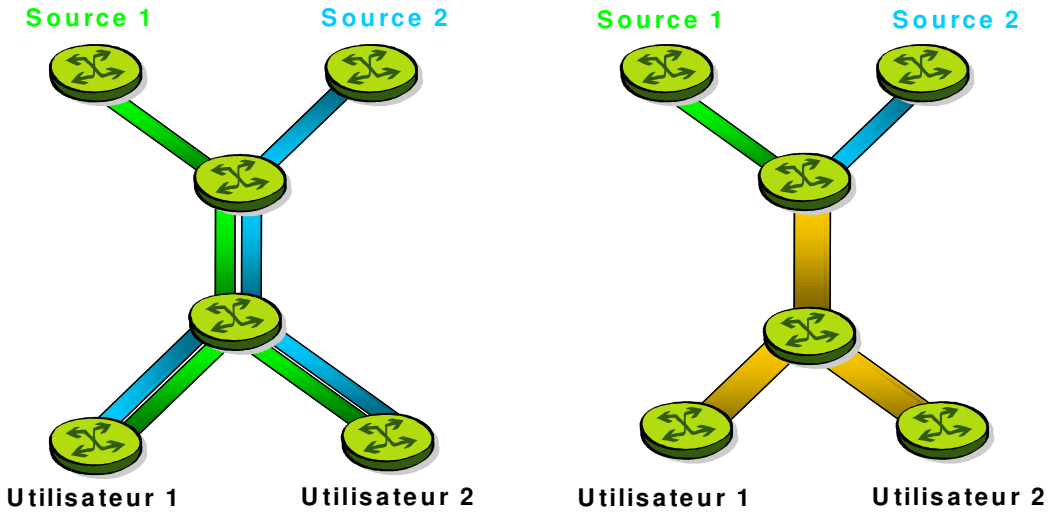


Figure 10: Stratégies d'agrégation d'arbre Multicast

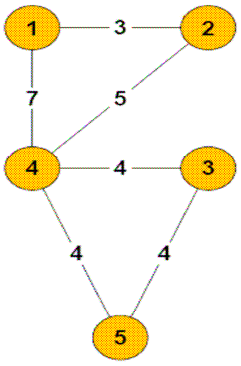
F. Formulation du problème de mono-routage des LSPs

F.1. Introduction

MPLS grâce à son Traffic Engineering permet de définir des chemins de routage explicites dans les réseaux IP (avec RSVP ou CR-LDP). C'est cette possibilité que nous allons exploiter pour fournir une solution au problème de mono-routage des LSPs dans les réseaux MPLS. En effet cette particularité permet de proposer un large panel d'algorithmes de routage dits «hors ligne» .

Dans ce qui suit nous allons illustrer le problème de mono-routage des LSPs sur un petit exemple.

Nous considérons le cas du réseaux suivant :



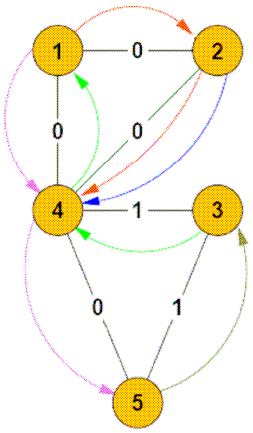
Flow	Demand	Path
1 ⇒ 5	4	
2 ⇒ 4	2	
5 ⇒ 3	3	
3 ⇒ 1	3	
1 ⇒ 4	3	

Figure 11: Exemple de réseau

Les valeurs sur les arcs représentent la bande passante de ceux-ci. Le tableau représente les demandes à mono-router.

La question que l'on se pose est comment faire passer cette demande sur le réseau tout en respectant les contraintes de capacité sur les arcs. Même pour un réseau aussi simple la solution n'est pas triviale. D'ailleurs la solution basée sur les plus courts chemins n'est pas réalisable dans ce cas.

Une solution possible est illustrée dans la figure suivante :



Flow	Demand	Path
1 ⇒ 5	4	1 4 5
2 ⇒ 4	2	2 4
5 ⇒ 3	3	5 3
3 ⇒ 1	3	3 4 1
1 ⇒ 4	3	1 2 4

Figure 12: Exemple de solution réalisable

Ainsi la bande passante résiduelle est de 2. Mais alors que dire de la qualité de cette solution ? Est-ce qu'il existe une solution qui dégage plus de bande passante résiduelle ?

Ces questions ont au moins le mérite de soulever une problématique intéressante. En effet un critère très important à intégrer lors de la recherche d'une solution au problème de mono-routage est la bande passante résiduelle. Car plus celle-ci est importante plus le réseau peu faire face aux évolutions possibles de la matrice des demandes (nouvelles demandes, pannes,…).

F.2. Approche basée sur l'aspect bande passante

Dans cette approche, le problème comme décrit précédemment consiste à affecter des ressources aux différents LSPs, tout en respectant les contraintes de capacité sur les arcs. Nous proposons dans ce qui suit une formulation basée sur l'optimisation de la bande passante.

Le problème peut être formulé de la façon suivante :

- ✓ Soit un réseau de *N* nœuds et *M* interfaces représenté par un graphe *G = [X,U], C_u* la bande passante de chaque interface *u* du réseau. Soit aussi *K* LSPs à placer. Chaque LSP est défini par sa source *s(k)* et sa destination *t(k)*, et finalement sa bande passante requise d^k.

- ✓ Trouver une et une seule route pour chaque LSP, qui satisfait et la contrainte de bande passante et les contraintes administratives (affinité, sécurité,…). Autrement dit, soit *p(k)* le nombre de chemins dans G ayant comme source *s(k)* et comme destination *t(k)*, soit alors P_j^k le j^{éme} chemin entre *s(k)* et *t(k)*. Pour tout *j =1…p(k),* le problème consiste à trouver un jeu de variables de mono-routage $x_j^k \in \{0,1\}$ tel que les contraintes de capacité sur les interfaces et de conservation des flots soient satisfaites :

$$\sum_{k=1}^{K}\sum_{j=1}^{p(k)} P_j^k(u)\cdot x_j^k \cdot d^k \le C_u \quad , \quad u = 1...M \tag{1.1}$$

et

$$\sum_{j=1}^{p(k)} x_j^k = 1 \tag{1.2}$$

avec

$$P_j^k(u) = \begin{cases} 1 \; si \; u \in P_j^k \\ 0 \; \; sinon \end{cases}$$

Soit $x^k = \left(x_j^k\right)_{j=1...p(k)}$ et $x = \left(x^k\right)_{k=1...K}$ le vecteur qui représente une solution du problème.

Considérons alors les variables $y_u^k = \sum_j P_j^k(u) \cdot x_j^k \cdot d^k \quad et \quad y_u = \sum_{k=1}^{K} y_u^k$.

y_u^k est la capacité utilisée par le LSP k sur l'interface *u*, y_u est la capacité utilisée par tous les LSP sur l'interface *u.*

Nous définissons comme le coût d'une solution x la valeur :

$$\Gamma(x) = \sum_{u=1}^{M} F(y_u, C_u) \tag{1.3}$$

La fonction *F* étant définie par :

$$F(y,C) = \begin{cases} A_1 \cdot y^2 & \text{si } y \le C \\ A_1 \cdot y^2 + A_2 \cdot (y-C)^2 & \text{si } y > C \end{cases} \tag{1.4}$$

avec A_1 et A_2 deux paramètres ajustables (A_2=10000*A_1).

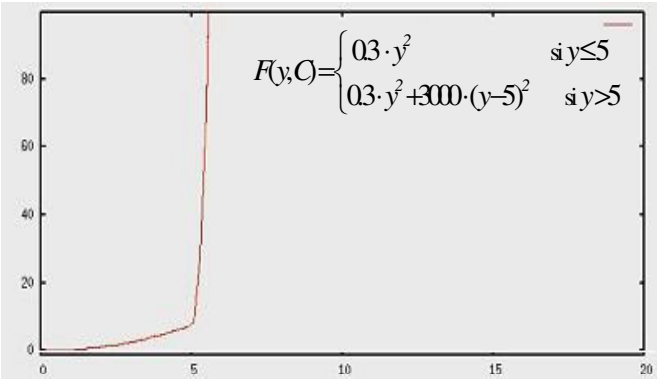


Figure 13: Fonction pénalité quadratique

La fonction F agit comme une pénalité sur la contrainte de capacité. Nous avons de plus choisie une pénalité non-linéaire car celle-ci assure qu'une solution admissible au sens des capacités est trouvée s'il en existe une et permet d'obtenir une solution de coût minimum même quand il n'en existe pas.

La fonction quadratique *F* permet de « s'affranchir » de la contrainte sur les capacités en pénalisant toutes les interfaces pour lesquelles il y a dépassement de capacité (leur coût devient très grand). Ainsi on peut garantir une bonne répartition du trafic dans le réseau et éviter les goulots d'étranglements. Bien entendu, le « réglage » d'une pénalité est délicat. Les coefficients présentés ci-dessus sont le résultat d'un important nombre de tests que nous ne détaillerons pas ici.

Le problème consiste donc à trouver une solution qui minimise $\Gamma(x)$ et satisfait la contrainte de mono-routage (1.2).

$$\begin{cases} Minimiser \ \ \Gamma(x) = \sum_{u} F(y_u, C_u) \\ sous \\ \sum_{j=1}^{p(K)} x_j^k = 1, k = 1...K \\ x_j^k \geq 0, \ j = 1...p(K), k = 1...K \end{cases} \quad (1.5)$$

G. Conclusion

Dans ce chapitre, nous décrivons le fonctionnement d'un réseau IP à travers les deux protocoles de routages les plus répandus (OSPF, BGP). Nous présentons les évolutions majeures d'IP pour la mise en place d'applications multiservices nécessitant certaines performances de qualité de service.

Nous définissons brièvement la norme MPLS ainsi que les mécanismes de routage par LSP. Nous introduisons le rôle du « Traffic Engineering » ainsi que la prise en charge du multicast au niveau des réseaux MPLS.

Finalement nous proposons une première modélisation du problème de mono-routage des LSPs dans les réseaux MPLS à travers une approche basée sur la bande passante.

« 640K, ce devrait être assez pour tout le monde. »

Bill Gates,
1981.

Chapitre 2 - LES RESEAUX MPLS ET GESTION DE LA QOS

A. Introduction

L'Internet de demain devra permettre le déploiement d'applications multimédia ayant des exigences spécifiques en terme de QoS. Certains services comme les services vocaux auront besoin d'un faible délai point à point et d'une faible gigue. D'autres, comme les trafics de données, nécessiteront des faibles taux de perte ou d'erreurs sans retransmission avec éventuellement une certaine garantie de bande passante pour le trafic de données transactionnel.

Pour pouvoir garantir la QoS des flux transportés, il va donc falloir utiliser des mécanismes permettant de traiter de manière différenciée les différentes catégories de trafic dans les organes du réseau ainsi que des protocoles de signalisation de la QoS pour pouvoir allouer des ressources en fonction des besoins des applications.

La QoS se décline principalement en quatre paramètres : débit, délai, gigue et perte. Le débit représente les ressources de transmission occupées par un flot. Un flot est un ensemble de paquets résultant d'une application utilisatrice. Le délai correspond au temps de transfert de bout en bout d'un paquet. La gigue correspond aux variations de latence des paquets. La gigue provient essentiellement des variations de trafic sur les liens de sorties des routeurs. Des pertes de paquets peuvent être dues à des erreurs d'intégrité sur les données ou des rejets de paquets en cas de congestion.

L'introduction de la Qualité de Service (QoS) permet à l'Internet de fournir d'autres classes de service que le traditionnel « best effort ». Pour cela il faut que le réseau soit capable d'isoler les flots pour leur fournir la QoS requise en leur offrant un traitement spécifique. Aujourd'hui, l'IETF défini deux approches de QoS : Services Intégrés (IntServ) [RFC1633] et Services Différenciés (DiffServ) [RFC2475].

Dans ce chapitre nous allons présenter les principaux mécanisme de gestion de la QoS dans les réseaux IP/MPLS. Nous allons par la suite présenter les modèles que l'on va utiliser pour représenter ces mécanismes. Ainsi qu'une modélisation du problème de mono-routage des LSPs qui intègre la prise en charge des indicateurs de QoS.

B. IntServ

Le modèle IntServ [RFC1633] définit une architecture capable de prendre en charge la QoS en définissant des mécanismes de contrôle complémentaires sans toucher au fonctionnement IP. C'est un modèle basé sur le protocole de signalisation RSVP [RFC2205].

Dans ce modèle, les routeurs réservent les ressources pour un flot de données spécifiques en mémorisant des informations d'état. Il est important de rafraîchir périodiquement les informations au cas où il y a eu changement de la route empruntée par le flot. En effet, il est inutile de continuer à réserver les ressources sur un routeur qui ne fait plus partie du chemin emprunté.

IntServ a défini trois types d'éléments réseau (NE:Network Elements):

- **QoS-capable-NE** : offrant un ou plusieurs services IntServ;
- **QoS-aware-NE** : supportant des services équivalent à ceux de IntServ
- **non-QoS-NE** : ne supportant pas les fonctionnalités IntServ;

IntServ définit deux nouveaux types de services:

- **Guaranteed Service (GS)** : qui garantit la bande passante et un délai d'acheminement limité (Audio, Vidéo). Pas de gestion de la gigue.

- **Controlled Load** : qui est équivalent à un service Best Effort dans un environnement non surchargé.

L'architecture IntServ repose sur deux principes fondamentaux :

- le réseau doit être contrôlé et soumis à des mécanismes de contrôle d'admission
- des mécanismes de réservation de ressources sont nécessaires pour fournir des services différenciés

Le protocole RSVP est utilisé pour signaler les exigences de QoS par flot aux éléments du réseau (hôtes, routeurs ou sous-réseaux). Les éléments du réseau, selon les ressources disponibles, implémentent l'un des services IntServ en fonction du type de la QoS souhaitée pendant la transmission des données.

Ainsi, globalement trois types de services sont définis en fonction principalement des besoins applicatifs vis-à-vis des délais, mais aussi, corrélativement par rapport au débit disponible.

	Débit	Délai	Gigue	Service
Applications élastiques	++	=	=	BE : Best Effort
Applications temps réel	+++	+	+	CL : Controlled Load
Applications temps réel interactives	+++	+++	+++	GS : Guarenteed Service

Tableau 1: Les services IntServ

Le best effort classique n'offre aucune garantie particulière et est donc assez bien adapté aux applications élastiques.

Une architecture telle que IntServ/RSVP qui requiert de maintenir des états *par flot* pose certains problèmes pour le « *passage à l'échelle* ». Une augmentation très importante du nombre de flots à gérer se traduit par un accroissement de la charge de traitement qui peut devenir insupportable.

Cette charge se situe aussi bien sur le plan des données, où des états de classification et d'ordonnancement doivent être maintenus par flot et manipulés à

chaque paquet, que sur le plan de contrôle où il faut maintenir des états de gestion par flot.

Les progrès réalisés au niveau des moteurs de commutation ne sont pas suffisants pour palier à une telle charge spécialement dans les cœurs de réseaux.

En effet au problème du facteur d'échelle vient s'ajouter le problème du haut débit qui nécessite des temps de traitements encore plus courts.

C'est pourquoi nous avons plutôt étudié une solution basée sur l'agrégation de trafics par classes.

C. DiffServ

Le principe de DiffServ [RFC2475] est de différencier des agrégations de flots regroupés par classes de services (temps réel, transactionnel ou « best effort » par exemple). Cela signifie que DiffServ n'utilise pas de mécanisme de réservation de ressources et offre aux flots une QoS située entre le « Best-Effort » (IP normal) et le « Hard QoS Guarantees ».

L'avantage de ce modèle est d'une part, sa proximité avec IP, ce qui permet une implantation aisée, et d'autre part ses capacités d'évolution face à l'introduction de nouveaux services.

Le modèle DiffServ apporte une QoS différenciée pour chaque classe de service selon un contrat prédéfini avec l'émetteur des flux de données. Ce contrat correspond à un ensemble de paramètres (bande passante garantie, pic de données acceptés, …) ainsi que les micro-flux associés à chaque classe de service.

Les flux dont les besoins sont similaires sont regroupés (agrégation), puis en fonction du champ DSCP, sont traités spécifiquement par les routeurs (Per Hop Behavior) : choix de file d'attente, algorithme de gestion, ...

Un domaine DiffServ définit un ensemble de nœuds réseau appliquant une politique de QoS commune. Dans un domaine DiffServ, les champs « TOS » (IPv4) ou « Traffic Class » (IPv6) des entêtes IP sont remplacés par un champ appelé DS dont la structure est la suivante :

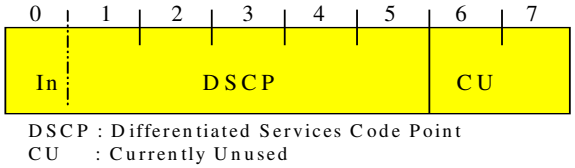


Figure 14: Structure du champ DS dans une entête IP

La signification des différents champs est la suivante :

- Le champ DSCP (DiffServ Code Point) indique la classe du paquet,
- Le champ in indique si le paquet est « In/Out profile » (peut être éliminé ou non),

- Le champ CU n'est pas utilisé pour l'instant.

C.1. Les composants du modèle DiffServ

On peut distinguer les routeurs situés dans le cœur de réseau « core routers » de ceux situés à sa frontière « edge routers ». Les routeurs du cœur de réseau du domaine DiffServ réalisent des opérations simples de bufferisation et d'avancement des paquets de chaque flot en se basant uniquement sur le marquage fait par les routeurs situés à la frontière du domaine DiffServ. Les « core routers » possèdent les deux mécanismes cruciaux du modèle DiffServ, le Scheduling et le Buffer Management. C'est à ce niveau que la différenciation de service est faite.

Les « edge routers » sont les portes d'entrée obligatoires pour un flot pénétrant dans le domaine DiffServ. Ils effectuent la classification, le metering, le marking, le shaping et le policing de chaque flot. Les traitements effectués par un edge router, décrits dans la figure ci-dessous, sont les suivants:

- **Classifier :** ce mécanisme permet de déterminer à quelle classe de service appartient un paquet. Cette classification peut être réalisée soit en examinant le champ TOS ou d'autres champs du paquet (adresses source et destination, protocole, ports, etc.). Le classifier détermine également si ce paquet peut accéder à cette classe en se basant sur le contrat (SLA: Service Level Agreement) passé entre l'utilisateur et l'opérateur.

- **Meter :** ce mécanisme mesure à la volée les caractéristiques des trafics de cette classe injectés par la source (débits moyen et maximal, longueur maximale d'un burst au débit maximal, etc.). Si ces caractéristiques dépassent le contrat passé avec l'opérateur (SLA), le Meter indique que le paquet est « out profile » en positionnant le champ « in/out ». Cela signifie que par la suite ce paquet pourra être traité en « best-effort » voire même supprimé en cas de congestion. Sinon le paquet est marqué "in profile".

- **Marker :** C'est à ce niveau qu'est réalisé l'agrégation des flots en classes. Le Marker détermine le PHB (Per Hop Behavior) du paquet, et en accord avec les informations transmises par le Meter, positionne le champ DSCP (marquage de la classe). Il est important de noter que cela n'est pas fait par le classifier, car un même flot suivant les conditions de trafic peut être marqué différemment.

- **Shaper :** Il régule les flots suivant les caractéristiques de leur classe. Le plus souvent cela est fait par la technique du token bucket; des utilitaires de fragmentation et de compression peuvent y être associés (LFI et RTP de CISCO).

- **Policer et dropper :** Si les paquets ne sont pas conformes, ils peuvent être jetés ou déclassés dès l'entrée du domaine DiffServ (dans le cœur du réseau, les paquets ne pourront plus être déclassés).

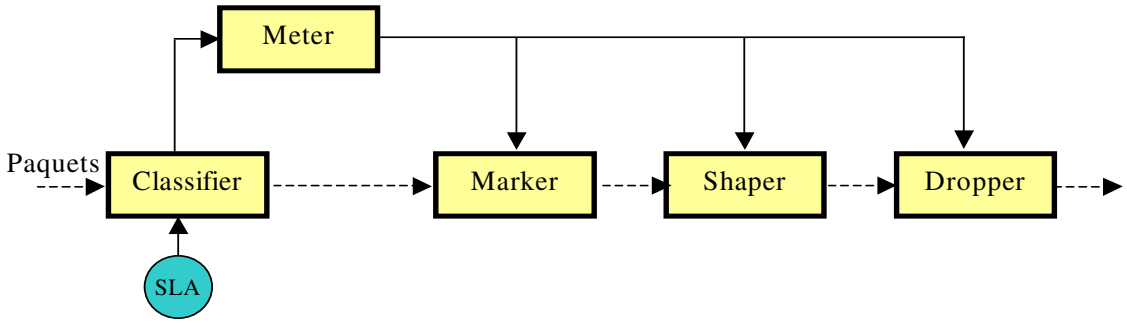


Figure 15: Traitement des flots dans un edge router

Dans un routeur DiffServ du cœur de réseau, chaque sortie possède un nombre fixe de files d'attente logiques où le routeur dépose les paquets arrivants sur la base de leur PHB. Les files d'attente sont servies en accord avec l'algorithme d'ordonnancement.

Le « core router » est constitué de trois éléments principaux. Le routage proprement dit qui consiste à la détermination du PHB, le scheduling et la gestion de buffer (cf. figure ci-dessous)

- **Routage :** Cette opération consiste à aiguiller le paquet vers un port de sortie et à déterminer son nouveau PHB.

- **Scheduling :** Plusieurs politiques d'ordonnancement peuvent être utilisées : WFQ, WRR ou priorités fixes. A l'heure actuelle, il semblerait que les opérateurs s'orientent vers une combinaison de ces politiques avec une priorité fixe pour le trafic temps réel et un ordonnancement WFQ pour les autres classes. C'est le champ DSCP des paquets qui permet d'affecter les paquets à une file d'ordonnancement particulière.

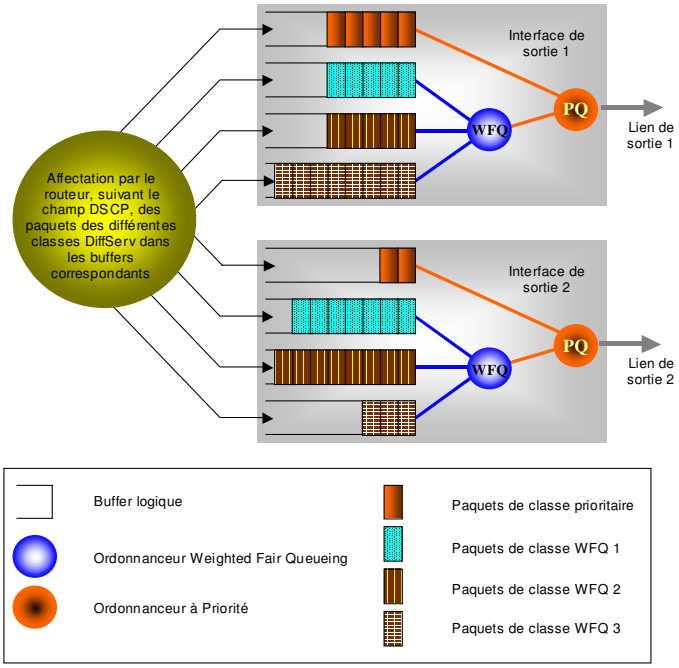


Figure 16: L'ordonnancement DiffServ dans un routeur IP/MPLS

Une des premières architectures proposées pour DiffServ est l'architecture TDSA (Two-bit Differentiated Service Architecture). Cette architecture comprend trois classes de service codées sur deux bits : Premium (10), Assured Forwarding (01), et Best Effort (00).

Le service Premium est destiné au trafic temps réel (voix en particulier). L'Assured Forwarding reçoit un meilleur traitement que le Best-Effort au niveau des pertes (gestion de buffer). Pour le service Premium les paquets en trop « out profile » sont systématiquement jetés au niveau du edge router. L'Assured Forwarding service a la garantie que si les flots restent dans le profil négocié, ils ne sont jetés qu'en dernière extrémité. Par contre, s'ils sont hors profil, ils peuvent être déclassés par le edge router.

D. MPLS et DiffServ

MPLS permet à l'administrateur réseau de spécifier comment les agrégats DiffServ (DiffServ Behavior Aggregates : BAs) sont affectés aux LSPs. En fait, la question est de savoir comment affecter un ensemble de BA au même LSP ou à des LSPs différents. Quand les paquets arrivent à un Ingress du domaine MPLS, le champ DSCP est ajouté à l'entête qui correspond au Behavior Aggregate (BA).

A chaque nœud traversé, c'est le DSCP qui va permettre de sélectionner le PHB (Per Hop Behavior), qui définit le Scheduling (classe prioritaire, WFQ etc.) et les probabilités de rejet du paquet. Cette correspondance entre les classes DiffServ et le PHB de MPLS offre de multiples possibilités à un administrateur pour un cœur de réseau DiffServ/MPLS.

L'affectation de classes DiffServ à différents LSPs permet aussi d'attribuer des mécanismes de protection différents pour ces différentes classes de service. Dans ce qui suit, nous décrivons les deux grands types de LSP que sont les E-LSP et les L-LSP. Un réseau MPLS peut combiner et faire cohabiter « à souhait » ces deux types de LSP.

D.1. E-LSP (EXP-InferredPSC LSPs)

Un LSP peut être utilisé pour transporter une ou plusieurs OA (Ordered Aggregate) ; c'est un ensemble de BA (Behavior Aggregate).

Un E-LSP peut être constitué de 8 BA pour une FEC donnée. Dans de tels LSPs, le champ EXP est utilisé pour déterminer le comportement PHB (informations d'ordonnancement et de rejet (dropping).

Le routeur LSR peut être configuré pour affecter automatiquement un PHB en fonction du champ EXP. Si cette configuration n'a pas été faite, le routeur prendra une préconfiguration par défaut pour tous les EXP vers le PHB par défaut. L'utilisateur peut aussi spécifier cette correspondance.

Plusieurs E-LSP peuvent être fusionnés dans un autre LSP à condition que ce LSP supporte le même ensemble de BA. Cette fusion est plus simple lorsque qu'on utilise une correspondance EXP-PHB préconfigurée (qui sera partout la même).

Par exemple, AF1x est un PSC (PHB Scheduling Class) comprenant les PHB de AF11, AF12 et AF13

D.2. L-LSP (Label-Only-Inferred-PSC LSPs)

Avec ce genre de LSP, les informations d'ordonnancement sont transportées implicitement dans le label. Le champ EXP contient les informations de perte uniquement. Dans chaque LSP de cette famille, un seul PSC et donc un seul OA est acheminé par LSP.

Dans ce cas aussi, les L-LSP peuvent être fusionnés dans un autre L-LSP, mais à la condition que le nouveau L-LSP supporte la même PSC.

Par exemple EF est un PSC comprenant exclusivement le PHB de EF.

E. Modélisation de la QoS dans les réseaux IP/MPLS

Nous avons vu que la QoS du réseau est conditionnée principalement par le débit offert, les délais des paquets à travers le réseau, les pertes et la gigue.

C'est pourquoi nous avons enrichi notre modèle bande passante en y intégrant la prise en compte de ces indicateurs. L'objectif d'une telle modélisation est de permettre une évaluation plus fine des critères de performance et de les intégrer dans un processus d'optimisation des réseaux IP/MPLS à qualité de service.

Nous présentons ainsi deux algorithmes d'ordonnancement de paquets utilisés dans l'architecture DiffServ. Nous présentons aussi l'extension QoS du problème de mono-routage des LSP.

E.1. Introduction du Modèle QoS

Maximiser la bande passante résiduelle est une bonne approche tant que le réseau n'est pas chargé. En effet l'aspect non-linéaire (des délais, pertes etc.) n'est pas primordial puisque toutes les contraintes de QoS seront satisfaites. Dans ce cas un algorithme qui « remplit » intelligemment le réseau sera suffisamment correct. Dans le cas où le réseau commence à être chargé, il en va tout autrement, et la recherche d'une solution à moindre coût satisfaisant les contraintes de QoS est un problème difficile qu'il faut traiter.

Ainsi on peut se demander quelle est la valeur de la QoS pour un LSP qui se trouve avec 50% d'occupation sur un lien? Cette valeur paraît confortable, pourtant elle peut donner une valeur de délai déjà trop élevée pour un lien. Sachant que le LSP utilisera plusieurs liens, la violation des contraintes sera d'autant plus forte.

Le modèle précédent peut être étendu pour prendre en compte les contraintes de QoS imposées pour les LSPs. Ces contraintes découlent des spécificités de chaque

LSP et surtout des trafics qu'il transporte (contraintes de délai de bout en bout, de pertes, de gigue). Dans ce qui suit, nous ne considérons que les deux premières contraintes (délai et pertes).

Les classes de services qui circulent sur notre réseau sont : EF, AF1x AF2x AF3x AF4x et Best Effort. Ces différentes classes sont affectées aux files d'attente correspondantes selon un mapping local à chaque interface. En effet, les paquets arrivent au niveau du routeur et sont classés selon leur label MPLS et/ou leur DSCP. C'est ce qui permet de les diriger vers les bonnes interfaces de sorties. Ensuite ces paquets sont ordonnancés selon la politique d'ordonnancement de la file (PQ ou WFQ). Pour les files gérées par WFQ (au nombre de 3 pour notre cas) nous définissons aussi les poids relatifs α_2, α_3, α_4.

Soit *c* = 1…4 les classes finales (après mapping) et soit *k* un LSP de classe *c*. Soit alors T^k le délai et B^k la probabilité de pertes du LSP *k*. Sous l'hypothèse d'indépendance, T^k et B^k peuvent être calculés à l'aide des formules suivantes :

$$T^k = \sum_j x_j^k \sum_{u \in P_j^k} T_u^c \quad et \quad B^k = 1 - \sum_j x_j^k \prod_{u \in P_j^k} (1 - B_u^c) \tag{2.1}$$

T_u^c : est le délai de la classe c au niveau de l'interface u.

B_u^c : est la probalité de pertes de la classe c au niveau de l'interface u

Les bornes de QoS sont associées aux services qui sont eux même associés à des classes Diffserv. Nous définissons $\overline{T}^c$ et $\overline{B}^c$ les bornes supérieurs pour le délai et la probabilité de pertes pour la classe *c* comme suit.

$$\overline{T}^c = Min(\overline{T}^s / s\ un\ service\ et\ classe(s) = c)\ ;\ \overline{B}^c = Min(\overline{B}^s / s\ un\ service\ et\ classe(s) = c)$$

Alors T^k et B^k doivent satisfaire les inégalités suivantes :

$$T^k \le \overline{T}^c \quad et \quad B^k \le \overline{B}^c \tag{2.2}$$

L'évaluation de T^k et de B^k est liée à la politique d'ordonnancement utilisée dans le réseau. Dans les réseaux actuels l'une des politiques les plus répandues est la politique FIFO (First In First Out) de part la simplicité de son déploiement. Celle-ci permet d'avoir une estimation grossière des délais et des pertes lorsque d'autres politiques d'ordonnancement sont employées. Dans ce qui suit nous proposons une modélisation de cette politique par les files M/M/1/N.

D'autres politiques sont de plus en plus usitées afin de garantir une QoS plus fine. Notamment les politiques à base de priorités, de partage des ressources ou encore de superposition de ces deux dernières. Nous proposons ainsi une modélisation plus complexe et plus fine de la QoS basée la politique LLQ qui est une superposition d'une file prioritaire et de 3 files WFQ.

E.1.1. Première formulation avec le modèle M/M/1/N

Dans les réseaux à commutations de paquets, certaines ressources peuvent avoir des capacités finies conduisant à des pertes de paquets. Le modèle M/M/1/N permet d'étudier de tels systèmes.

Pour ce modèle l'idée est de supposer une file d'attente globale par interface. Les paquets qui arrivent sur cette file d'attente sont traités par la politique d'ordonnancement FIFO.

L'ordonnancement FIFO sert les paquets dans leur ordre d'arrivée dans le routeur. De ce fait, il ne fournit aucune différentiation de service, mais reste extrêmement simple et rapide. Le délai subi par les paquets est le même pour toutes les classes (une seule file d'attente).

La file M/M/1/N est un système à capacité finie, avec N-1 positions d'attente. Les clients arrivent selon un processus de Poisson de taux λ. Ils sont servis suivant la discipline FIFO et la durée du service est une variable aléatoire exponentiellement distribuée de moyenne $1/\mu$.

Les clients qui à leur arrivée trouvent un système plein sont rejetés et n'ont plus aucune influence sur le système. Ainsi la capacité finie agit comme un régulateur sur le nombre de clients en attente, et par conséquent aucune hypothèse sur le taux d'utilisation $\rho = \lambda / \mu$ n'est nécessaire.

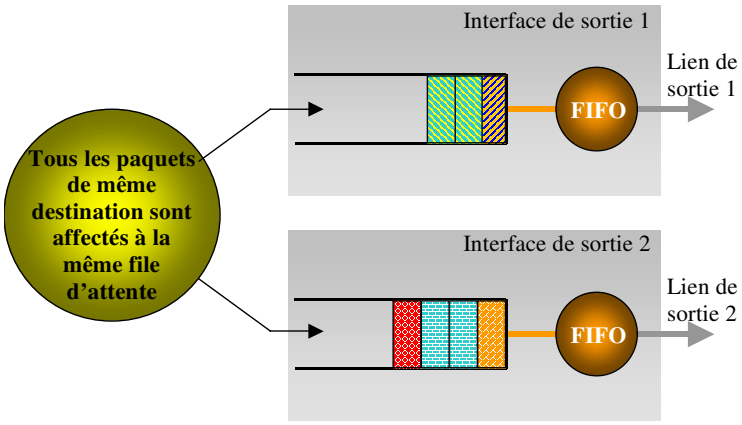


Figure 17: Ordonnancement FIFO dans un routeur

Ainsi au niveau de chaque interface on peut calculer la charge de chaque file d'attente (ayant une taille de buffer égale à *N*) à l'aide de la formule non linéaire suivante :

$$X_u(\infty) = \begin{cases} \dfrac{N\rho^{N+2} - (N+1)\rho^{N+1} + \rho}{1 - \rho - \rho^{N+1} + \rho^{N+2}} & si\ \rho \neq 1 \\ \dfrac{N}{2} & si\ \rho = 1 \end{cases} \qquad (2.3)$$

La probabilité de pertes au niveau d'une interface *u* s'exprime de la façon suivante :

$$P_u(N) = \begin{cases} 1 + \dfrac{(1-\rho^N)}{(1-\rho^{N+1})} & si\ \rho \le 1 \\ \dfrac{1}{N+1} & si\ \rho = 1 \end{cases} \quad (2.4)$$

ρ étant le taux d'utilisation de cette file d'attente.

Le calcul du délai au niveau de l'interface *u* se fait alors de la façon suivante :

$$T_u = \frac{X_u(\infty)}{\lambda \cdot (1 - P_u(N))} \quad (2.5)$$

En utilisant ce modèle nous avons décidé de prendre le délai et les pertes comme critère d'optimisation. Ainsi le problème (1.5) se formule dans ce cas de la façon suivante :

$$\begin{cases} Minimiser\ \ \Gamma(x) = \sum_{j=1}^{K} \sum_{u \in P_j^k} \alpha \cdot T_u + \beta \cdot P_u(N) \\ sous \\ \sum_{j=1}^{p(K)} x_j^k = 1, k = 1...K \\ x_j^k \ge 0, j = 1...p(K), k = 1...K \end{cases} \quad (2.6)$$

$\alpha\ et\ \beta$ sont deux paramètres qui permettent d'équilibrer l'importance des deux critères dans le processus d'optimisation.

C'est un modèle qui s'est révélé être efficace pour le problème du routage des LSPs, à la seule condition d'être sûr, au préalable, qu'une solution admissible existe.

La formule du délai entraînant des débordements numériques ($\rho \approx 1 \Rightarrow P_u(N) \approx 1$) il est difficile d'utiliser ce modèle pour les cas où la solution ne serait pas admissible (manque de bande passante, taille de buffer trop petite, etc.).

E.2. Les files à priorités

L'introduction de mécanismes de gestion de priorités dans les réseaux IP est relativement récente. Elle est due à un besoin grandissant de différenciation de service au sein du réseau IP. Des constructeurs comme CISCO proposent maintenant plusieurs algorithmes d'ordonnancement dans leurs routeurs. Le plus simple est l'ordonnancement FIFO.

Le traitement par files à priorités (PQ) est un type de traitement permettant de faire de la différentiation de service. Les paquets qui entrent dans le système sont tout d'abord classifiés et envoyés sur la file d'attente correspondant à la classe de priorité

à laquelle ils appartiennent. Les paquets contenus dans la file prioritaire sont servis (envoyés sur le port de sortie) en premier, quand cette file est vide les autres files peuvent être servies.

Il y a un intérêt évident pour une file à priorité par exemple pour traiter le trafic de contrôle du réseau. En cas de congestion ou de problème dans le réseau, il est primordial que le trafic de reconfiguration des protocoles puisse circuler entre les routeurs, quelque soit la charge des routeurs.

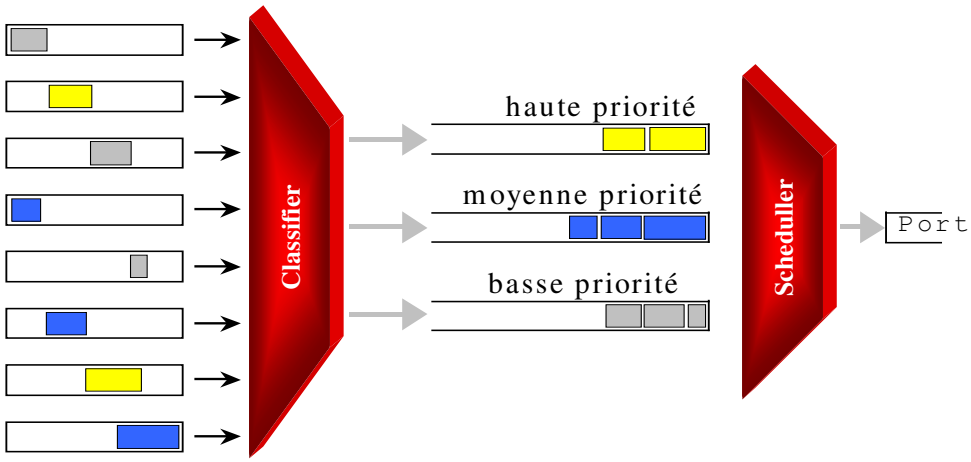


Figure 18: Principe du traitement des paquets par priorités (PQ)

Dans ce qui suit nous présentons une modélisation des systèmes à priorités dans le cas de k classes.

On suppose un ordre de priorités pour les classes

Priorité(Classe 1) > Priorité(Classe 2) >…. >Priorité(Classe k)

Nous présentons d'après [KLE76] les formules du délai et nombre moyen de clients. Avec les notations suivantes :

- $\rho_k = \frac{\lambda_k}{\mu_k}$ le taux d'utilisation associé à une classe k,
- T_k le temps de séjour moyen d'un client de classe k dans le système,
- W_k le temps de séjour moyen d'un client de classe k dans la file d'attente,
- N_k l'espérance du nombre de clients associés à la file k présents dans le système,
- X_k l'espérance du nombre de clients associés à la file k présents dans la file d'attente,
- $W_0 = \sum_{j=1}^{k} \frac{\lambda_k \cdot \bar{\sigma}_j^{\,2}}{2}$ le temps de service résiduel moyen d'un client dans le serveur.

Ce système est ergodique et présente une solution quand

$$\rho = \sum_{j=1}^{k} \rho_j \prec 1$$

Le temps de séjour moyen d'un client de classe k dans la file d'attente k est donné par :

$$w_k = \frac{W_0}{\left(1 - \sum_{j=1}^{k-1} \rho_j\right) \cdot \left(1 - \sum_{j=1}^{k} \rho_j\right)}$$

Ce qui nous donne un temps de séjour moyen dans le système de :

$$T_k = \frac{1}{\mu_k} + \frac{W_0}{\left(1 - \sum_{j=1}^{k-1} \rho_j\right) \cdot \left(1 - \sum_{j=1}^{k} \rho_j\right)}$$

En appliquant la loi de Little, nous obtenons le nombre moyen de clients dans le système pour la classe k :

$$N_k = \rho_k + \frac{\lambda_k \cdot W_0}{\left(1 - \sum_{j=1}^{k-1} \rho_j\right) \cdot \left(1 - \sum_{j=1}^{k} \rho_j\right)}$$

E.3. Les files WFQ

Les algorithmes à partage de bande passante sont tous basés sur GPS (Generalized Processor Sharing) qui est un ordonnancement idéal non implantable directement où les flux sont vus comme des fluides. L'objet de ce modèle est de garantir une bande passante minimum à chaque classes de trafic (DiffServ) et ceci à tout instant.

Autrement dit, il permet de garantir un débit minimum à chaque classe. Pour ce faire, les classes sont servies en parallèle avec une portion du débit de l'interface proportionnelle à des poids. La bande passante minimum garantie pour chaque classe est alors la fraction de débit de la ligne qui leur est allouée lorsque toutes les classes sont continuellement présentes.

Plusieurs algorithmes ont été proposés pour simuler GPS, dont les plus connus sont WFQ et WF^2Q (Worst-case Fair Weighted Fair Queuing) [GUE99], [GOL94] et [BEN96]. Ils peuvent être vus comme des algorithmes à priorités dynamiques. Grâce à l'isolation des files d'attente, ces algorithmes peuvent offrir des garanties pour toutes les classes de service en termes de délai et de débit.

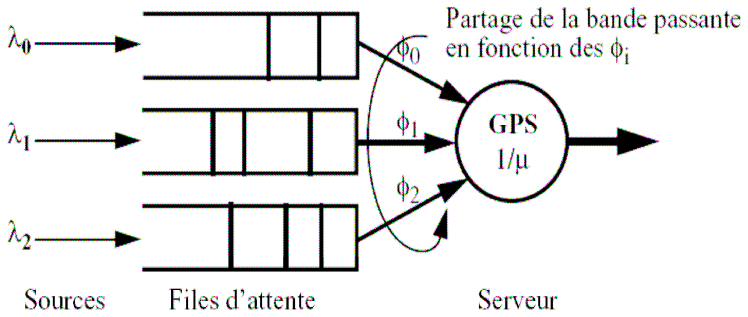


Figure 19: Ordonnancement WFQ sur 3 files

L'analyse d'un tel système est complexe car le taux de service de chaque flux dépend de l'état de la file (présence ou non d'un flux). L'évaluation des valeurs stationnaires exactes a été discutée dans [ADA00] et [BOX03]. L'obtention de solutions analytiques pour un nombre quelconque de classes semble peu évidente. Un algorithme itératif a été proposé par [TAN00] permettant d'approcher le taux de service équivalent pour chaque classe.

Une amélioration a été proposée pour cette approche [BRU02], en utilisant une décomposition en file M/H/1. Toutefois, ces approches sont basées sur des algorithmes itératifs qui peuvent devenir lourds pour de grands réseaux. De plus, elles sont peu aisées pour l'optimisation et le dimensionnement des réseaux.

L'approche retenue ici utilise le comportement quasi linéaire des algorithmes de types GPS lorsque la charge du système n'est pas trop élevée.

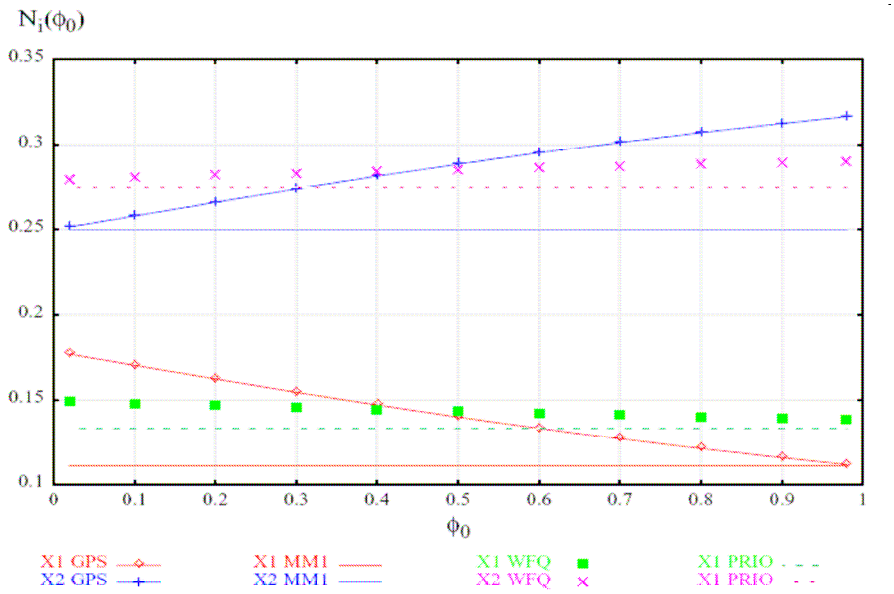


Figure 20: Linéarité GPS et WFQ en fonction des pondérations

La première observation faite est la quasi linéarité du nombre de clients par rapport à la pondération pour des services GPS ou WFQ. Une étude plus fine [BOC05] a

montré que lorsque le facteur d'utilisation globale est faible, le nombre de clients de chaque classe varie linéairement en fonction des poids ϕ_k .

La seconde observation concerne la différence entre l'ordonnancement idéal GPS et son émulation par WFQ. En effet, GPS étant idéal, lorsque le poids d'une classe tend vers un, celle-ci est complètement isolée alors que pour WFQ, du fait de la granularité des paquets, ce n'est pas le cas. La classe tendra à être simplement prioritaire. Nous utiliserons ces observations pour approximer le comportement WFQ dans ce qui suit.

Contrairement aux mécanismes à priorités, WFQ tente (théoriquement du moins) de donner un pourcentage de la bande passante à des classes de service. En fait, au lieu de compter le nombre de paquets servis et de faire un service de type Round Robin pondéré sur les paquets, WFQ va faire de manière infinitésimale un Round Robin sur les bits d'un paquet.

Dans WFQ, c'est donc bien la longueur du paquet qui va être comptée comme le service donné à une classe de trafic et non le nombre de paquets. Et comme la longueur du paquet est bien une partie de la bande passante consommée, il en résulte que c'est bien la bande passante qui est partagée (et pondérée) entre les classes de service. WFQ tend en moyenne à faire ce qu'aurait fait un algorithme Round Robin Pondéré si on avait pu traiter les messages bit par bit.

En général WFQ est implémenté dans les routeurs par logiciel, ce qui limite un peu le principe dans le nombre de file etc. Le nombre de files étant limité, il sera donc impossible de traiter chaque flot individuellement pour lui garantir une certaine qualité de service.

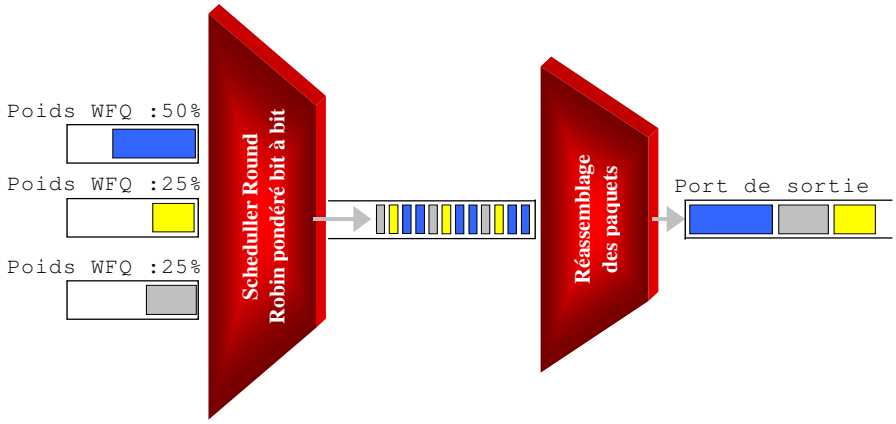


Figure 21: Principe idéalisé du traitement des paquets par WFQ

E.4. Deuxième formulation avec le modèle LLQ

E.4.1. Validation du modèle LLQ

Nous formulons le problème précédent (2.6) dans le cadre d'un réseau MPLS-Diffserv. Les interfaces des routeurs possèdent chacune 4 files d'attente. Une des

files d'attente est gérée par une politique d'ordonnancement « Priority Queuing » (PQ) les trois restantes sont gérées par « Weighted Fair Queuing » (WFQ).

La superposition des deux modèles de files d'attente (1 file prioritaire + 3 files WFQ) est proposée par Cisco sous le nom de LLQ [Cisco LLQ]. L'objectif est d'offrir un délai minimum pour des trafics critiques (files d'attente prioritaire), tout en ayant un partage des ressources pour les autres trafics.

La file d'attente prioritaire est servie comme dans l'algorithme de priorité fixe. Les files d'attente à partage de bande passante sont alors considérées comme une seule file ayant la plus petite priorité. Lorsque la file prioritaire est servie, l'algorithme WFQ est appliqué aux files d'attente restantes.

La modélisation de la QoS que nous adoptons dans ce qui suit repose donc sur cette superposition de modèles de files d'attente.

Nous allons donc dans ce qui suit nous intéresser aux délais et pertes induits par chacune de ces files d'attente. Ces deux critères constituent la base de l'évaluation de la partie QoS de notre fonction coût.

Nous définissons les variables suivantes :

- ✓ $y_u^c = \sum_{k/c(k)=c} \sum_{j/u \in P_j^k} x_j^k \cdot d^k$ la bande passante occupée par la classe *c* qui passe sur l'interface *u*.
- ✓ $y_u = y_u^1 + y_u^2 + y_u^3 + y_u^4$ la bande passante totale occupée sur l'interface *u*.
- ✓ z_u^c le nombre moyen de paquets de la classe *c* sur l'interface *u*
- ✓ $z_u = z_u^1 + z_u^2 + z_u^3 + z_u^4$ la charge totale (en paquets) de l'interface *u*.

Des mesures faites dans le cadre du projet RNRT ESQUIMAUX [ESQ02], montrent que la superposition de plus de 15 trafics (voix, vidéo…) peut être approximée par un trafic poissonien (utilisation nominale ie. < 60%). Cette approximation peut être donc utilisée dans le cadre d'un cœur de réseau où l'agrégation de trafics dépasse largement les 15 trafics.

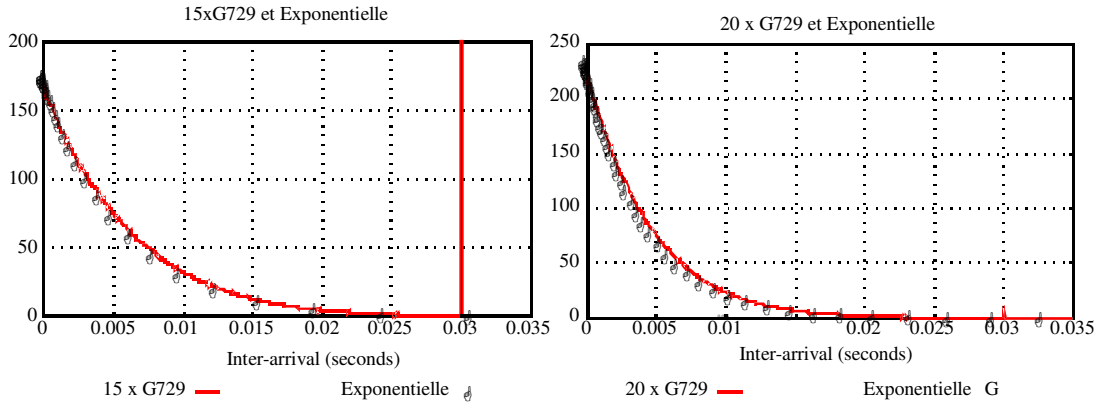


Figure 22: Superposition de sources ON-OFF (G729)

La figure 22 représente l'évolution de la distribution d'un ensemble de sources G729 en fonction de leur nombre. Nous pouvons constater que plus ce nombre est grand, plus la courbe tend vers une distribution exponentielle.

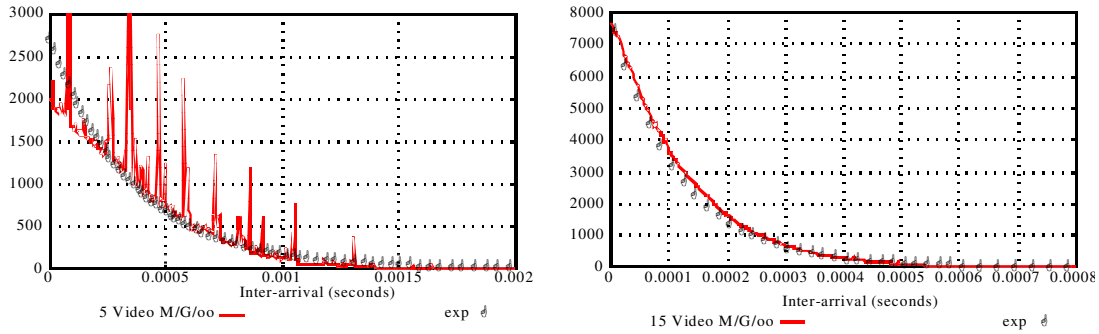


Figure 23: Superposition de sources vidéo (M/G/∞)

La figure 23 montre la distribution des inter-arrivées issues de la superposition de 5 et 15 sources vidéo, comparées avec les distributions exponentielles respectivement de même débit moyen. Comme dans le cas des « codecs voix » de type ON-OFF, la distribution d'inter-arrivées d'une superposition de sources tend rapidement (à partir de 10 sources) vers la distribution exponentielle de même moyenne.

Cette propriété est bien vérifiée dans les cœurs de réseau (MPLS par exemple) qui agrègent de nombreux trafics.

C'est pourquoi nous allons utiliser une modélisation des files d'attente de type M/M/1/N pour évaluer z_u^c et z_u, nous pourrons par la suite en déduire T_u^c.

D'un autre côté pour le calcul de la probabilité de perte, une approximation grossière consiste à supposer pour chaque interface *u* une taille de buffer globale N_u.

$$B_u^c = B_u = (1-\rho)\cdot\rho^{N_u}/(1-\rho^{N_u+1}) \quad \forall c \quad avec \quad \rho = y_u / C_u$$

Cette approximation qui suppose la même probabilité de blocage pour chacune des classes est justifiée par le fait qu'il n'existe à ce jour aucune formule analytique exacte de cette probabilité pour chacune des classes.

Soit alors $\tilde{\rho}_u^1 = y_u^1 \cdot (1-B_u)/C_u \quad et \quad \tilde{\rho}_u^i = y_u^i \cdot (1-B_u)/C_u^r \quad i = 2,3,4$

$avec\ C_u^r = C_u - y_u^1$ la bande passante résiduelle pour les classes WFQ.

Soit aussi $\tilde{\rho}_u = \sum_{i=2}^{4} \tilde{\rho}_u^i$, nous utilisons les variables intermédiaires suivantes :

$$\tilde{Z}_u^{2,3,4} = \tilde{\rho}_u^1/(1-\tilde{\rho}_u^1) \quad , \quad \tilde{Z}_u^{2,3} = (\tilde{\rho}_u^2+\tilde{\rho}_u^3)/(1-\tilde{\rho}_u^2-\tilde{\rho}_u^3)$$
$$\tilde{Z}_u^{2,4} = (\tilde{\rho}_u^2+\tilde{\rho}_u^4)/(1-\tilde{\rho}_u^2-\tilde{\rho}_u^4) \quad , \quad \tilde{Z}_u^{3,4} = (\tilde{\rho}_u^4+\tilde{\rho}_u^3)/(1-\tilde{\rho}_u^4-\tilde{\rho}_u^3) \qquad (2.7)$$

Le délai de la classe prioritaire (1) au niveau de l'interface *u* s'exprime donc comme suit :

$$T_u^1 = z_u^1 / y_u^1 \quad avec \quad z_u^1 = \tilde{\rho}_u^1 /(1-\tilde{\rho}_u^1) \tag{2.8}$$

Les délais pour les classes WFQ (2, 3,4) au niveau de l'interface *u* sont alors :

$$\begin{aligned} T_u^2 &= z_u^2 / y_u^2 \quad avec \quad z_u^2 = \alpha_2 \cdot \tilde{\rho}_u^2 /(1-\tilde{\rho}_u^2) + (1-\alpha_2)\cdot[\tilde{Z}_u^{2,3,4} - \tilde{Z}_u^{3,4}] \\ T_u^3 &= z_u^3 / y_u^3 \quad avec \quad z_u^3 = \alpha_3 \cdot \tilde{\rho}_u^3 /(1-\tilde{\rho}_u^3) + (1-\alpha_3)\cdot[\tilde{Z}_u^{2,3,4} - \tilde{Z}_u^{2,4}] \\ T_u^4 &= z_u^4 / y_u^4 \quad avec \quad z_u^4 = \alpha_4 \cdot \tilde{\rho}_u^4 /(1-\tilde{\rho}_u^4) + (1-\alpha_4)\cdot[\tilde{Z}_u^{2,3,4} - \tilde{Z}_u^{2,3}] \end{aligned} \tag{2.9}$$

L'idée exploitée dans cette formulation est la linéarité observée précédemment (Fig. 20). Ainsi la formule des z_u^i (i=2,3,4) peut être vue comme une interpolation linéaire entre une file M/M/1 est une file prioritaire.

Ainsi quand le poids de l'une des files tend vers 1 elle se comporte comme une seconde file prioritaire. Le système traite donc tous les paquets de celle-ci avant de traite les deux files restantes.

la simulation de ce modèle donne les résultats suivants :

ρ = 0,4	*Paramètres*		Service Exponentiel		
Queue	Débit	poids	Charge analytique	Charge simulation	Erreur relative (%)
Priorité	0.1		0.144	0.144	0.00
WFQ1	0.1	1	0.174	0.171	1.754
WFQ2	0.1	1	0.165	0.168	1.786
WFQ3	0.1	1	0.182	0.183	0.546
Global	0.4		0.665	0.666	0.150

Tableau 2: Comparaison du modèle et de la simulation pour *ρ = 0,4*

L'hypothèse de linéarité de la charge en fonction des pondérations sur laquelle est basée l'approximation étant particulièrement vérifiée lorsque la charge globale du système est faible, les résultats pour $\rho = 0.4$ sont excellents. Les erreurs relatives (en %) sont très faibles (<2%).

ρ = 0,55	*Paramètres*		Service Exponentiel		
Queue	Débit	poids	Charge analytique	Charge simulation	Erreur relative (%)
Priorité	0.2		0.337	0.337	0.00
WFQ1	0.2	4	0.477	0.490	2.65
WFQ2	0.1	2	0.268	0.257	4.28
WFQ3	0.05	1	0.140	0.135	3.70
Global	0.55		1.222	1.219	0.25

Tableau 3: Comparaison du modèle et de la simulation pour *ρ = 0,55*

Les résultats obtenus pour une charge totale de 0.55 sont encore très bons. Les erreurs relatives (en %) sont très faibles (<5%).

ρ = 0,8	*Paramètres*		Service Exponentiel		
Queue	Débit	poids	Charge analytique	Charge simulation	Erreur relative (%)
Priorité	0.3		0.643	0.643	0.00
WFQ1	0.2	1	1.33	1.36	2.206
WFQ2	0.1	1	0.697	0.648	7.562
WFQ3	0.2	1	1.33	1.358	2.062
Global	0.8		4.00	4.009	0.224

Tableau 4: Comparaison du modèle et de la simulation pour $\rho = 0,8$

Pour une charge globale de 0.8, les résultats sont un peu moins précis, car l'approximation linéaire devient grossière. Toutefois, Ils restent corrects (< 8%), notamment pour une évaluation rapide de bout en bout du délai.

Nous concluons que même pour des charges assez élevées le modèle simule parfaitement le fonctionnement de l'algorithme LLQ. C'est donc ce modèle que nous utiliserons pour évaluer les critères de QoS dans la boucle d'optimisation.

F. Formulation mathématique

Une formulation du problème de mono-routage des LSPs serait donc

$$\begin{cases} Minimiser \;\; \Gamma(x) = \sum_{u=1}^{M} F(y_u, C_u) \\ sous \\ T_k^c \le \overline{T}^c \qquad \forall c \in \{1..4\} \quad \forall k \\ B_k^c \le \overline{B}^c \qquad \forall c \in \{1..4\} \quad \forall k \\ \sum_{j=1}^{p(K)} x_j^k = 1, k = 1...K \\ x_j^k \ge 0, \; j = 1...p(K), k = 1...K \end{cases} \tag{2.10}$$

Ce problème est un problème combinatoire en variables binaires avec une fonction objectif non linéaire et des contraintes non linéaires. Cette formulation est difficilement exploitable. En effet les contraintes de QoS sont difficiles à gérer dans un processus d'optimisation. C'est pourquoi nous avons relaxé ces contraintes à l'aide de fonctions de pénalités quadratiques. Celles-ci vont permettre d'assurer les contraintes de QoS pour chaque LSP.

Nous avons testé plusieurs types de fonctions pénalités dont les pénalités exactes. Nous avons retenu le type quadratique car c'est un bon compromis entre rapidité et efficacité (à s'éloigner des contraintes).

Le problème précédent devient

$$\begin{cases} Minimiser \;\; \Gamma(x) = \sum_{u=1}^{M} F(y_u, C_u) + \sum_{k=1}^{K} D_{c(k)}(T^{c(k)}, \overline{T}^{c(k)}) + \sum_{k=1}^{K} L_{c(k)}(B^{c(k)}, \overline{B}^{c(k)}) \\ sous \\ \sum_{j=1}^{p(K)} x_j^k = 1, k = 1...K \\ x_j^k \geq 0, \; j = 1...p(K), k = 1...K \end{cases} \tag{2.11}$$

Avec pour c = 1..4, la fonction de pénalité sur les délais s'exprime par :

$$D_c(T^c, \overline{T}^c) = \begin{cases} \phi_1 \cdot (T^c)^2 & si \; T^c \leq \overline{T}^c \\ \phi_1 \cdot (T^c)^2 + \phi_2 \cdot (T^c - \overline{T}^c)^2 & sinon \end{cases} \tag{2.12}$$

et la fonction de pénalité sur les pertes par :

$$L_c(B^c, \overline{B}^c) = \begin{cases} \varphi_1 \cdot (B^c)^2 & si \; B^c \leq \overline{B}^c \\ \varphi_1 \cdot (B^c)^2 + \varphi_2 \cdot (B^c - \overline{B}^c)^2 & sinon \end{cases} \tag{2.13}$$

$\phi_1, \phi_2, \varphi_1, \varphi_2$ sont des paramètres ajustables qui permettent de donner plus de poids à un critère par rapport à un autre.

G. Conclusion

Dans ce Chapitre nous présentons les différents mécanisme utilisés pour la gestion de la QoS au niveau des réseaux MPLS.

Nous présentons une première formulation simple et pertinente du problème de mono-routage des LSPs à l'aide des files d'attente de type M/M/1/N.

Nous présentons par la suite une modélisation plus fine de la QoS par le biais de l'algorithme LLQ proposée par Cisco. Ce modèle combine une file prioritaire et trois files WFQ. Cette modélisation à été validée grâce à des résultats de simulations pour différentes valeurs de ρ (charge du système).

Nous proposons ensuite une deuxième formulation (plus précise) du problème de mono-routage des LSPs qui prend en compte les contraintes de QoS au niveau de chaque LSP.

« Il n’y a aucune raison valable pour que quiconque
ait envie d’avoir un ordinateur chez lui »

Ken Olsen président, fondateur de Digital Equipement Corp.,
1977.

Chapitre 3 - METHODES D'OPTIMISATION NON LINEAIRE

A. Introduction

Comme nous l'avons vu dans les chapitres précédent nous nous proposons de résoudre le problème de mono-routage des LSPs dans les réseaux MPLS. Ces problèmes sont réputés être NP-complet donc essayer de les résoudre directement serait une tâche très ardue. C'est pourquoi nous avons pensé à exploiter le savoir dont on dispose pour résoudre les problèmes de multi-routage. Nous utiliserons par la suite la solution multi-routée afin d'en déduire la solution de notre problème initial.

Ce chapitre présente donc un comparatif de toutes les méthodes que nous avons étudiés pour résoudre le problème multi-routé des LSPs. Le but étant d'en garder la plus efficace. Nous proposons aussi dans ce chapitre une nouvelle méthode ILSP (Iterative Loading Shortest Path) pour résoudre ce problème multi-routé.

Un grand nombre de problèmes d'optimisation peuvent être formulés sous la forme

$$(P) \quad \begin{cases} Min\ f(x) \\ x \in C \end{cases} \tag{3.1}$$

Les méthodes d'optimisation non linéaire permettent d'apporter des solutions itératives pour résoudre numériquement ce type de problèmes.

Celles-ci ont largement été traitées dans des ouvrages comme [MIN83], [LUE73], nous en aborderons quelques unes que nous avons jugées plus appropriées à la résolution de notre problème.

Nous rappelons que nous nous intéressons plus particulièrement au cas du problème de routage par partage de charge optimal. Il s'agit de trouver les coefficients de partage de charge optimaux $(\tilde{\alpha}_i)_i$ pour chaque demande. Ces coefficients définissent la manière dont est partagée la demande sur chacune des routes possibles.

En particulier nous allons traiter le cas des méthodes de type « gradient » pour la résolution des problèmes sans contraintes ainsi que les méthodes « Flow Deviation » et gradient projeté pour le cas des problèmes avec contraintes.

En fin de chapitre nous montrerons des résultats concernant la comparaison de ces algorithmes sur des réseaux tests.

B. La Méthode « Déviation de flots »

En 1973, Fratta, Gerla et Kleinrock [FRA73] proposaient la méthode de Déviation de flots pour traiter des problèmes de dimensionnement et routage dans les réseaux de communications. La méthode de déviation de flot permet de résoudre le problème de multiflot non contraint avec pour fonction objectif à minimiser une fonction non-linéaire différentiable convexe et séparable sur les arcs.

Cette méthode est une application de la méthode de Frank-Wolfe au problème de routage. Le principe de l'algorithme est celui des algorithmes de descente : on détermine une direction de descente et on effectue une minimisation unidimensionnelle dans cette direction. La recherche d'une direction de descente se ramène ici à une recherche de plus courts chemins. À chaque itération, on va ainsi dévier une certaine quantité de flot des routes existantes vers les plus courts chemins qui définissent la direction de descente.

Ces problèmes d'affectation de flots sur un réseau avec plusieurs paires origines et destinations sont encore très étudiés aujourd'hui pour leur complexité, essentiellement due aux aspects suivants :

- ✓ Les modèles font apparaître des multiflots qui induisent une compétition entre les différents flux origine-destination pour l'utilisation des bandes passantes disponibles ;

- ✓ Les critères de routage visent à minimiser la congestion du réseau, et à maximiser sa Qualité de Service (QoS), ce qui induit généralement des fonctions non linéaires ;

- ✓ Le nombre de variables est une fonction des dimensions du réseau et du nombre de paires origine-destination, conduisant à des modèles de grande taille ;

- ✓ L'hétérogénéité du trafic et la nécessité de le sécuriser face aux éventuelles pannes induisent des contraintes supplémentaires sur le choix des routes supportant les flux.

L'ouvrage de Bertsekas et Gallager [BER92] donne une description plus détaillée de ces problèmes et de leurs difficultés algorithmiques.

Appliquées aux problèmes de routage dans les réseaux de communication la méthode de « Déviation de flots » n'est autre qu'une variante de l'algorithme de linéarisation de Frank et Wolfe pour les problèmes de multiflots à coûts convexes. Elle a été proposée pour la première fois par Fratta, Gerla et Kleinrock [FRA73] dans le cadre de la conception de réseaux à commutation de paquets. Elle a été appliquée à la fois sur des problèmes de transport et des problèmes de télécommunication.

On rappelle le principe de l'algorithme de Frank-Wolfe pour les programmes non-linéaires avec contraintes linéaires.

Il s'agit de résoudre des problèmes de type :

$$(P) \quad \begin{cases} Min \ \ \Phi(x) \\ sous \ \ A \cdot x = b \\ x \geq 0 \end{cases} \tag{3.2}$$

où $\Phi(x)$ est une fonction non linéaire différentiable.

On suppose $\Omega = \{x \in R^n | A \cdot x = b, x \geq 0\}$, polyèdre des solutions réalisables, borné.

L'algorithme procède par résolutions successives de linéarisations locales de *(P)*. A l'itération *t* on détermine le gradient :

$$g^t = \nabla\Phi\left(x^t\right) \tag{3.3}$$

Au point courant x^t puis on résout le problème suivant

$$\left(P^t\right) \quad \begin{cases} Min \;\; g^t \cdot y \\ sous \;\; A \cdot y = b \\ y \geq 0 \end{cases} \tag{3.4}$$

Soit y^t la solution optimale de P^t. C'est un point extrême de Ω (puisqu'on suppose qu'il est borné). Le nouveau point x^{t+1} est solution d'une optimisation linéaire sur le segment $[x^t, y^t]$ qui s'écrit de la façon suivante :

$$Min_{\alpha \in [0,1]} \Phi\left(x^t + \alpha \cdot (y^t - x^t)\right) \tag{3.5}$$

Cet algorithme est connu pour offrir une convergence sous-linéaire [BER92]. La convergence est généralement rapide dans les premières itérations, lorsque le point courant est éloigné des points extrêmes. Lorsqu'on s'approche de l'enveloppe de Ω et si la solution optimale se situe sur une face du polyèdre, l'angle entre deux points extrêmes consécutifs tend à s'aplatir progressivement et la trajectoire se met à zigzaguer (cf. figure (24)).

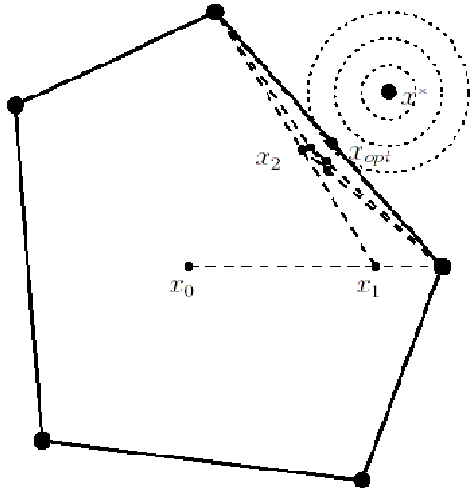


Figure 24: Phénomène de zigzag dans Frank-Wolfe

Le problème est séparable selon les flots et les deux étapes de l'algorithme sont les suivantes :

- Recherche du point extrême : pour chaque flot $k \in K$, déterminer un plus court chemin (o_k, d_k) au sens des dérivées premières de la fonction coût $\Phi(x)$. Soit $\tilde{p}_k$ ce chemin, le routage unique sur chacun des plus courts chemins constitue un point extrême du polyèdre des flots. On considère $\tilde{x}$ la combinaison de ces K chemins.

- Recherche linéaire du nouveau point : fixer à q_k la quantité de flot sur $\tilde{p}_k$ (le flot k est routé intégralement dessus) et rechercher le taux de reroutage optimal, c'est-à-dire le scalaire θ^* tel que :

$$\theta^* = \mathrm{Argmin}_{\theta \in [0,1]} \Phi\left(x^t + \theta \cdot \left(\tilde{x} - x^t\right)\right) \tag{3.6}$$

La recherche du pas optimal θ^* se ramène à une minimisation unidimensionnelle classique.

On modifie ensuite le flot sur chaque chemin p_k d'un ratio identique θ^* :

$$x_p^{t+1} = x_p^t + \theta^* \cdot \left(\tilde{x}_{\tilde{p}_k} - x_p^t\right), \ \forall k \in K, \ \forall p \in P^k \tag{3.7}$$

On «dévie» ainsi à chaque itération une proportion égale pour toutes les commodités des flots circulant sur des chemins non optimaux vers les chemins de longueur minimale par rapport aux dérivées premières des fonctions $\Phi(x)$. L'algorithme converge vers un minimum global du problème initial dans le cas convexe

Pour chaque flot, on intègre le nouveau chemin dans la liste des chemins actifs si ce dernier n'y figure pas déjà :

$$P^k = P^k \cup \{\tilde{x}_k\}, \ \forall k \in K \tag{3.8}$$

« Déviation de flots » est une méthode d'amélioration. A ce titre, elle nécessite qu'on lui fournisse une solution initiale réalisable, ce qui se révèle problématique dans le cas de problèmes fortement contraints. Un moyen de contourner la difficulté consiste à démarrer avec la solution fournie, qu'elle soit réalisable ou non, puis à tenter régulièrement durant le processus d'optimisation de restaurer sa réalisabilité (au sens des contraintes de capacités).

L'intérêt de l'algorithme de Frank-Wolfe pour les problèmes de multiflots réside d'une part dans sa simplicité et d'autre part dans son interprétation en sous-problèmes élémentaires : la résolution du sous-problème se traduit par la recherche d'un plus court chemin et l'obtention du nouveau point correspond à un reroutage équitable

d'une partie du flot des chemins actifs sur le nouveau chemin, d'où le nom de déviation de flots.

Classiquement on choisit la fonction M/M/1 comme fonction coût pour $\Phi(x)$.

Nous avons choisi de démarrer l'algorithme avec comme solution initiale les PCC au sens de la métrique (1/Bande passante résiduelle) afin d'assurer au maximum la faisabilité de cette solution initiale au sens des contraintes de capacités.

L'algorithme de « Déviation de flot » reste aujourd'hui, trente ans après la thèse de M. Gerla, une méthode très efficace et versatile pour traiter les problèmes de routage avec contraintes [MAH06]. C'est pourquoi nous avons décidé d'intégrer cette méthode dans notre étude à des fins comparatives.

C. La méthode ILSP (Iterative Loading Shortest Path)

L'algorithme ILSP est une contribution nouvelle parmi les algorithmes de génération de chemins optimaux. Les prémices de cette méthode ont vus le jour lors de mon stage au LAAS en 2000 [RAC00], elle fut présentée pour la première fois en 2002 [BRU02]. Nous avons présenté une version améliorée de cet algorithme en 2003 [RAC03]. Le principe de l'algorithme a aussi été utilisé lors de la thèse de [ROM03] sous le nom de « millefeuille ». L'objectif d'ILSP est de générer pour chaque LSP l'ensemble des « n » meilleurs chemins candidats.

Le principe d'ILSP est de diviser la matrice des demandes en LSPs entre les nœuds dorsaux en autant de couches que voulue par le concepteur, chaque couche représentant un tantième de la matrice initiale. On projette alors les couches une par une sur le graphe de réseau. Chaque couche de la matrice est parcourue et projetée intégralement, et pour chaque couple (nœud source, nœud destination), on projette cette portion de flux sur le plus court chemin (selon la fonction coût utilisée).

La projection de chaque part de flux charge graduellement le réseau, ce qui modifie pour chaque couche successive, le plus court chemin choisi. On évite alors de réutiliser les liens les plus chargés. Le nombre de couches va définir le nombre de projections et les proportions de flux de chaque projection.

Le problème résolu peut être modélisé sous la forme d'un problème de minimisation :

A l'itération k on résout le sous problème :

$$(P) \quad \begin{cases} \underset{\Delta x^k}{Min} \; f\left(x^k + \Delta x^k\right) \\ A \cdot \left(x^k + \Delta x^k\right) = 0 \end{cases} \tag{3.9}$$

avec A étant la matrice des contraintes du problème.

En théorie ce problème n'est pas simple à résoudre ; heureusement quand on l'applique au cas du problème de routage par partage de charge dans les réseaux de télécommunication, cette complexité disparaît. Cela revient à trouver à chaque

itération et pour chaque LSP le plus court chemin qui va occasionner la plus petite augmentation du coût de la solution. Ainsi, à chaque itération, nous allons propager un quantum du LSP sur un chemin minimisant la fonction coût globale adoptée.

ILSP charge donc progressivement le réseau avec des fractions de LSPs en *R* itérations (*R* étant le nombre maximum de chemins permis par LSP).

A chaque itération *j*, un chemin candidat P_k est calculé pour chaque LSP *k.* Au début de chaque itération *j,* nous définissons $\Omega_j = \{k \mid k = 1...K\}$ et pendant chaque itération *j* les opérations suivantes sont répétées *K* fois (pour tous les LSPs non marqués):

- ✓ Choisir un LSP $k \in \Omega_j$ selon une loi uniforme, router la demande (d^k/R) sur son plus court chemin $P_{j^*}^k$ au sens de la métrique $\Gamma(x)$ d'affinité compatible et marquer le LSP,
- ✓ Faire $P_k := P_k \cup \{P_{j^*}^k\}$ et $\Omega_j = \Omega_j - \{k\}$,
- ✓ Mettre à jour la charge de toutes les interfaces appartenant à $P_{j^*}^k$ en propageant la fraction de LSP (d^k/R),
- ✓ Enlever la marque du LSP *k.*

On va distinguer 2 phases dans la recherche de routes en fonction du nombre de couches :

- ✓ Avec un faible nombre de couches, chaque nouvelle projection de flux va ouvrir une nouvelle route sur le réseau, la réutilisation d'une route existante sera rare,
- ✓ Au delà d'une certaine valeur, les couches successives vont modifier les proportions des flux sur chacune des routes ouvertes lors des projections précédentes en privilégiant les plus courts chemins au sens de la métrique choisie.

C.1. Sur la convergence de l'algorithme ILSP

Le but de ce paragraphe est de montrer l'efficacité de l'heuristique ILSP à générer les bons chemins ainsi que les bons partages de charges.

Pour l'algorithme ILSP le quantum propagé est différent pour chaque LSP

$$Quantum(LSP_i) = \frac{d_i}{NbRouteMaxPerLsp} \tag{3.10}$$

NbRouteMaxPerLsp : est une constante fixée à l'avance qui reflète le nombre de route maximum permis par LSP (10 routes par LSP).

Comme on a pu le constater dans le cas ou les demandes sont disparates en taille, les « grosses demandes » sont privilégiées quant au choix des routes, elles s'accaparent ainsi plus de « bons » chemins que les « petites demandes ». Ce qui dans certains cas nous éloigne de la solution optimale.

ILSPv2 est un amélioration de l'algorithme ILSP de base. Dans cette version nous prenons un quantum unique pour tout les LSPs. Ainsi à chaque itération le même quantum de trafic est propagé pour chaque LSP est cela jusqu'à ce que tous les LSPs soient entièrement propagés.
Nous prenons pour quantum dans cette méthode la valeur suivante :

$$Quantum(LSP_i) = \frac{\underset{i}{Min}(d_i)}{NbRouteMaxPerLsp} \tag{3.11}$$

l'algorithme se déroule de la manière suivante:

- tant que tout les LSPs ne sont pas complètement propagés ($d_k = 0$)
  - ✓ Choisir un LSP $k \in \Omega_j$ selon une loi uniforme, router le *quantum* sur le plus court chemin $P_{j^*}^k$ au sens de la métrique $\Gamma(x)$ d'affinité compatible et marquer le LSP,
  - ✓ Faire $P_k := P_k \cup \left\{P_{j^*}^k\right\}$ et $\Omega_j = \Omega_j - \{k\}$, $d_k = d_k - quantum$,
  - ✓ Mettre à jour la charge de toutes les interfaces appartenant à $P_{j^*}^k$ en propageant la fraction de LSP (*quantum*),
  - ✓ Enlever la marque du LSP *k.*

Nous allons nous intéresser au cas de la topologie «Topo3» décrite dans la partie tests de ce chapitre. Nous allons étudier l'évolution du coût obtenue par ILSPv2 en fonction de la taille du quantum :

Taille quantum	*Coût de la solution*
24 kb	**9,16688**
12 kb	**9,1507**
6 kb	**9,14533**
3 kb	**9,1407**
1.5 kb	**9,13715**
750 b	**9,13849**
375 b	**9,13805**
188 b	**9,13784**
94 b	**9,13771**
47 b	**9,13766**
24 b	**9,13764**
12 b	**9,13762**

Tableau 5: Evolution du coût en fonction de la taille du quantum

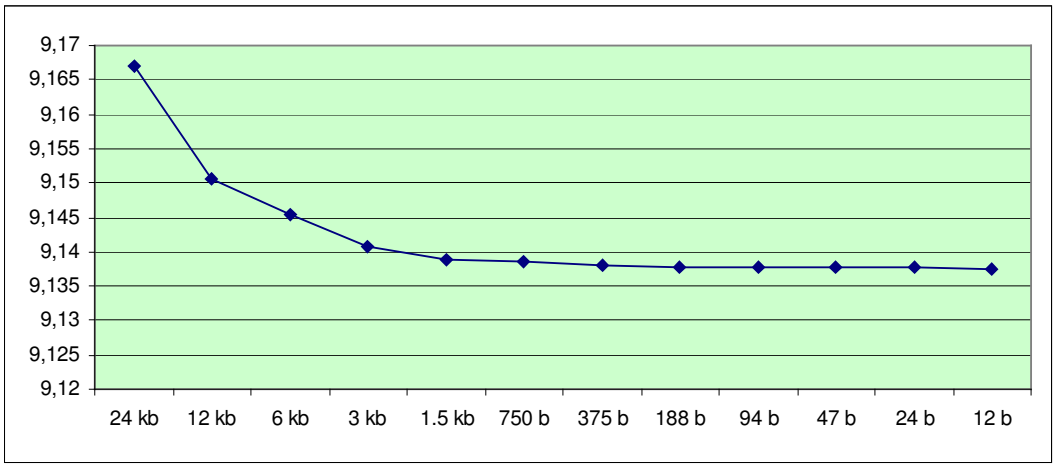


Figure 25: Evolution du coût en fonction de la taille du quantum

La courbe nous montre l'évolution de la solution fournit par l'heuristique. Plus le quantum est petit plus celle-ci est proche de l'optimum. Le minimum obtenu pour cette exemple se situe à 2,36% de l'optimum obtenu par la méthode « Gradient Projeté » (cf. tableau comparaison des coûts paragraphe E2).

L'inconvénient de cette approche réside dans le grand nombre d'itérations dans le cas d'un faible quantum. Cela se produit quand les demandes sont très disparates en tailles.

D. La méthode du gradient

Soit à résoudre un problème de type :

$$(P) \quad \begin{cases} Min \ f(x) \\ x \in \Re^n \end{cases} \tag{3.12}$$

Comme la stationnarité de *f* est toujours une condition nécessaire d'optimalité, pratiquement toutes les méthodes d'optimisation sans contraintes consistent à chercher un point stationnaire X* tel que $(\nabla f(X^*) = 0)$
Ce problème est équivalent à résoudre le système d'équations non linéaires :

$$\frac{\partial f}{\partial x_i}(x) = 0, \quad \forall i = 1, \ldots, n \tag{3.13}$$

On peut chercher à résoudre directement ce système, cependant cela suppose la fonction deux fois différentiable et nécessite le calcul des dérivées secondes en chaque point.

Les problèmes d'optimisation non linéaire, en général, ne permettent la détermination du gradient de *f* que numériquement. C'est pourquoi nous nous sommes limité aux méthodes du premier ordre. Les plus couramment utilisées consistent en des procédures itératives où l'on engendre une suite de points (u_k) convergeant vers un optimum local de *f*.

Pour construire une suite d'approximations d'un minimum d'une fonction *f*, on peut procéder de la façon suivante : au point u_k, on se donne une direction de descente d_k $\left(\nabla f(X^*)\cdot d_k \prec 0\right)$, et on cherche $\rho(u_k, d_k)$ tel que

$$f\left(u_k + \rho\left(u_k, d_k\right)\cdot d_k\right) = \inf_{\rho \in \Re} f\left(u_k + \rho \cdot d_k\right) \tag{3.14}$$

On définit alors la suite

$$u_{k+1} = u_k + \rho\left(u_k, d_k\right)\cdot d_k \tag{3.15}$$

Comme le gradient indique localement la direction de la plus forte augmentation de *f*, on prend pour direction de descente :

$$d_k = -\nabla f(u_k) \tag{3.16}$$

Le pas de déplacement est $\rho(u_k, d_k)$ suivant cette direction.

L'inconvénient de cette méthode est sa vitesse de convergence qui est faible. Son intérêt est de pouvoir se généraliser au cas des fonctions non partout différentiables convexes en utilisant la notions de sous gradient.

E. La méthode du gradient conjugué

Une grande liberté existe pour le choix des directions de descente, l'idée du gradient conjugué est de construire progressivement des directions mutuellement conjuguées. L'objectif est ici d'utiliser plus d'information sur *f* pour calculer la direction de descente qu'avec la méthode du gradient, sans pour autant être conduit au coût élevé de la méthode de Newton, qui nécessite de calcul de dérivées secondes. Le principe de la méthode consiste à chercher u_{k+1} tel que

$$f\left(u_{k+1}\right) = \inf_{v \in G_k} f\left(u_k + v\right) \tag{3.17}$$

Avec

$$G_k = \left\{ \sum_{i=1,k} \alpha_i \cdot \nabla f\left(u_i\right); \alpha_1, \ldots, \alpha_k \in \Re \right\} \tag{3.18}$$

Dans le cas d'une fonction quadratique :

$$F(X) = \frac{1}{2}\cdot X^T \cdot A \cdot X + B^T \cdot X + C \tag{3.19}$$

Les directions $d_1 \ldots d_k$ sont conjuguées par rapport à la matrice A

$$d_i^T \cdot A \cdot d_j = 0 \qquad \forall i \neq j; \quad i, j = 1,\ldots,k \tag{3.20}$$

en notant $g_k = \nabla F(X_k)$, la méthode prend la forme suivante :

Soit X_0 le point initial, $d_0 = -g_0$;
A l'étape k :

$$X_{k+1} = X_k + \alpha_k \cdot d_k \quad \text{avec } \alpha_k = \frac{g_k^t \cdot d_k}{d_k^t \cdot A \cdot d_k} \tag{3.21}$$

et

$$d_{k+1} = -g_{k+1} + \beta_k \cdot d_k \text{ avec} \qquad \beta_k = \frac{g_{k+1}^T \cdot d_k}{d_k^T \cdot A \cdot d_k} \tag{3.22}$$

On démontre que les directions ainsi engendrées sont mutuellement conjuguées [MIN83].

F. La méthode de Fletcher Reeves

Bien que la méthode de Fletcher-Reeves soit généralement classée parmi les méthodes du deuxième ordre, elle ne nécessite pas l'évaluation du Hessien de la fonction.

La méthode de Fletcher-Reeves [FLE64] est une extension de la méthode du gradient conjugué au cas des fonctions quelconques.

Elle présente l'avantage de converger plus rapidement que les méthodes de gradient classiques et de nécessiter beaucoup moins de calculs et de place mémoire que les méthodes quasi-newtoniennes.

On peut résumer simplement son principe :

Soit X_0 le point de départ, on pose $d_0 = -\nabla F(X_0)$

A l'étape k, on choisit α_k minimisant :

$$g(\alpha) = F(X_k + \alpha \cdot d_k) \tag{3.23}$$

On calcul

$$X_{k+1} = X_k + \alpha_k \cdot d_k \tag{3.24}$$

Et

$$d_{k+1} = -\nabla F(X_{k+1}) + \beta_k \cdot d_k \tag{3.25}$$

Avec

$$\beta_k = \frac{\left\|\nabla F\left(X_{k+1}\right)\right\|^2}{\left\|\nabla F\left(X_k\right)\right\|^2} \tag{3.26}$$

Pour assurer la convergence globale de la méthode, il est nécessaire de procéder à une réinitialisation périodique. On peut par exemple, toutes les n itérations, repartir du point courant avec, comme direction de déplacement, le gradient en ce point.

G. La méthode de Broyden, Fletcher, Goldfarb, Shanno (BFGS)

Cet algorithme fait parti des méthodes quasi-newtoniennes [BRO70],[FLE70], [GOL70], [SHA70] dont le principe général consiste en une généralisation de la formule itérative de Newton :

$$X_{k+1} = X_k - \alpha_k \left[\nabla^2 f(X_k)\right]^{-1} \nabla f(X_k) \tag{3.27}$$

Où l'on remplace $\left[\nabla^2 f(X_k)\right]^{-1}$ par une matrice définie positive H_k donnant la direction de déplacement à partir du gradient. La direction de descente de la méthode quasi-newton BFGS est définie par

$$d_k = -H_k \cdot \nabla f(Xk) \tag{3.28}$$

La matrice H_k est modifiée à chaque itération de manière à converger vers l'inverse du Hessien (pour une fonction quadratique, cela est vrai proche de la solution). Pour une fonction quelconque on obtient une approximation définie positive de l'inverse du Hessien de f.

Parmi toutes les variantes, la formule itérative la plus connue actualisant H_k est la suivante :

$$H_{k+1} = H_k + \left(1 + \frac{\gamma_k^T H_k \gamma_k}{\delta_k^T \gamma_k}\right) \cdot \frac{\delta_k \delta_k^T}{\delta_k^T \gamma_k} - \frac{\delta_k \gamma_k^T H_k + H_k \gamma_k \delta_k^T}{\delta_k^T \gamma_k} \tag{3.29}$$

avec :

$$\delta_k = X_{k+1} - X_k \tag{3.30}$$

et :

$$\gamma_k = \nabla f(X_{k+1}) - \nabla f(X_k) \tag{3.31}$$

Pour une fonction quelconque (non quadratique), il est nécessaire de réinitialiser H_k toutes les n itérations, comme pour les gradients conjugués.

H. La méthode du gradient projeté

Pour adapter les méthodes d'optimisation sans contraintes aux problèmes avec contraintes, une idée assez naturelle consiste à projeter à chaque itération le déplacement sur la frontière du domaine. Ainsi le gradient projeté revient à cheminer le long de la frontière dans la direction de la plus forte pente « relative » c'est à dire permise par les contraintes. L'un des premiers algorithmes bâtis suivant ce principe est le gradient projeté de Rosen [ROS60].

Cette méthode est d'autant plus intéressante quand les contraintes sont linéaires. Ainsi pour un problème de la forme suivante :

$$\begin{cases} Min \;\; F(x) \\ sous \;\; a_i \cdot x \le b_i \quad \forall i \in I_1 \\ \quad\;\; a_i \cdot x = b_i \quad \forall i \in I_2 \\ \quad\quad x \ge 0 \end{cases} \tag{3.32}$$

Soit alors à l'étape k, $I_k = \{i / a_i . X_k = b_i\}$ l'ensemble des indices des contraintes saturées, on cherche une direction d_k ($\|d_k\| = 1$) qui rende $\nabla F^T \cdot d_k$ minimal et qui satisfasse les relations :

$$a_i \cdot d_k = 0, \;\; \forall i \in I_k \tag{3.33}$$

La solution explicite de ce problème est obtenue en projetant le vecteur $-\nabla F(X)$ sur le sous espace $S_k = \{y / A_k \cdot y = 0\}$ ou A_k est la sous matrice de *A* constituée par les lignes $i \in I_k$ (On fait l'hypothèse que A_k est de rang plein)

On obtient $d_k = \dfrac{\bar{y}}{\|\bar{y}\|}$ avec $\bar{y} = -P_k \cdot \nabla F(X_k)$ où P_k est la matrice de projection donnée par :

$$P_k = I - A_k^T \cdot \left\{ A_k \cdot A_k^T \right\}^{-1} \cdot A_k \tag{3.34}$$

La détermination du déplacement α_k doit bien sûr se faire en tenant compte des contraintes. Une fois obtenue la direction de déplacement, on détermine le déplacement maximal autorisé $\alpha_{\max}$ tel que :

$$\alpha_{max} = \max_{\alpha} \left(\alpha / \alpha \ge 0; \; X_k + \alpha \cdot d_k \in D \right) \tag{3.35}$$

Où :

$$D = \left\{ X / X \in \Re^n ; a_i \cdot X \le b_i \;\; pour \; i = 1 \; à \; m \right\} \tag{3.36}$$

Le pas de déplacement est alors obtenu en résolvant le problème :

$$\min_{0 \le \alpha_k \le \alpha_{\max}} \left\{ F\left(X_k + \alpha_k \cdot d_k\right) \right\} \tag{3.37}$$

Soit $u = -\left[A_k \cdot A_k^T\right]^{-1} \cdot A_k \cdot \nabla F(X)$, si lorsque $\bar{y} = 0,\ u \geq 0$ alors on a atteint un minimum local de la fonction (les conditions de Kuhn et Tucker sont satisfaites). Dans le cas contraire on supprime dans I_k la contrainte pour laquelle la composante u_i de u est la plus négative ce qui permet d'obtenir une nouvelle direction de déplacement.

Contrairement à ce qu'on a vu pour la méthode de « Déviation de flots » qui se base sur la résolution de sous problème à coût linéaire, la méthode du gradient projeté, se base sur la résolution de sous problèmes à coût non-linéaire (quadratique). Ce sont des problèmes plus difficiles à résoudre mais la convergence est meilleure que pour le « Déviation de flots ».

L'algorithme du gradient projeté se déroule de la façon suivante :

- Soit x_0 une solution initiale, $x_0 \in X$ (ensemble des solutions admissibles)
- Pour k = 0,….
  - Si x_k est stationnaire STOP.
  - Sinon $x_{k+1} = x_k + \alpha_k \cdot \left(y_k - x_k\right)$
- FIN

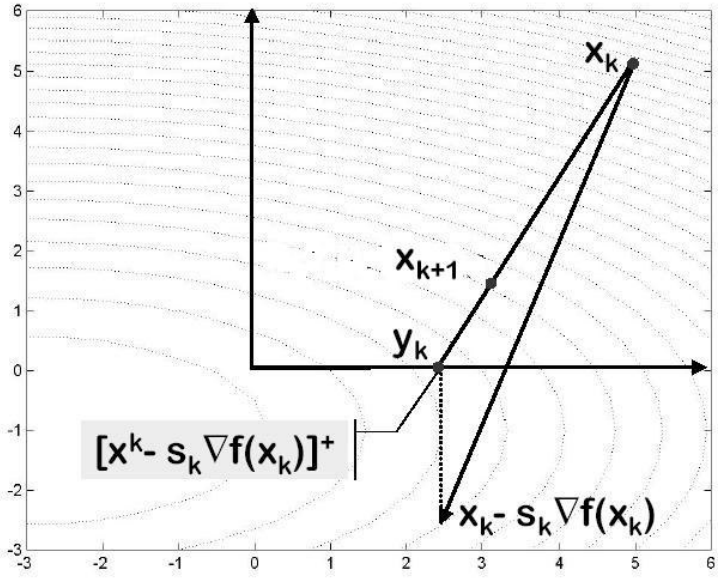


Figure 26: Construction des points x_k par la méthode du gradient projeté

- Avec $\alpha_k \in]0,1]$
- $s_k \in \Re, s_k \succ 0$
- y_k est la projection du point $\left(x_k - s_k \cdot \nabla f\left(x_k\right)\right)$

Ceci dit la méthode n'est efficace que si la projection est suffisamment simple.

I. Tests et résultats sur l'optimisation non linéaire

I.1. Introduction

Nous rappelons le contexte de ces tests. Il s'agit de comparer les différentes méthodes citées auparavant par rapport au problème du routage par partage de charge.

Certaines de ces méthodes ne sont pas totalement adaptée à la résolution de ce problème. Notamment pour les méthodes sans contraintes nous projetons les directions de descente sur le domaine des solutions réalisables. Nous fixons aussi la liste des chemins candidats pour chaque LSP.

Afin d'illustrer les performances des différentes méthodes citées dans ce chapitre nous allons utiliser quatre réseaux tests de différentes tailles.

Les tailles des problèmes abordés sont décrites dans le tableau suivant :

	#Sommets	***#Liens***	***#LSP***	***#Solutions***
Topo1	**6**	**8**	**5**	**768**
Topo2	**6**	**9**	**22**	**8.16293e+10**
Topo3	**15**	**29**	**42**	**1.89622e+18**
Topo4	**21**	**33**	**832**	**2.52498e+105**

Tableau 6: Caractéristiques des topologies de test

Le problème que nous allons aborder dans les tests qui suivent est le problème du routage en partage de charge optimal, c'est à dire que pour un ensemble de routes fixé, quelle est la répartition optimale du LSP sur chacune des routes possibles ?

Nous comparerons les méthodes selon 3 critères, à savoir le coût de la solution obtenue, le nombre d'itérations et le temps d'exécution.

Nous utilisons comme tests d'arrêts les conditions suivantes :

- $\|\nabla \Gamma\| \prec 10^{-6}$ (la norme du gradient)
- Si pendant 10 itérations aucune amélioration relative n'est obtenue $(\frac{coût(iter\ k) - coût(iter\ k-1)}{coût(iter\ k)} \prec 10^{-6})$
- Nombre d'itérations maximum fixé à 10000 itérations

I.2. Comparaison des coûts des solutions

Nous prenons comme fonction coût la fonction de pénalité suivante :

$$F(y,C) = \begin{cases} A_1 \cdot y^2 & \text{si } y \le C \\ A_1 \cdot y^2 + A_2 \cdot (y-C)^2 & \text{si } y > C \end{cases}$$

avec y la charge du lien et C sa capacité.

Coût	Topo1	Topo2	Topo3	Topo4
Ospf	4,35594e+5	1,33e+7	12,3389e+7	9,29762e+8
ILSP	2,13264e+5	1,1259e+7	9,33302e+7	7,06758e+08
ILSP v2	2,12518e+5	1,1223e+7	9,16688e+7	7,06696e+8
Flow Deviation	2,12147e+5	1,1185e+7	9,12969e+7	7,03125e+8
Fletcher Reeves	2,19460e+5	1,1097e+07	8,97234e+07	6,95267e+08
BFGS	2,11574e+5	1,1082e+07	8,92597e+07	6,94115e+08
Gradient Projeté	2,11574 e+5	1,1083e+07	8,92598e+07	6,94115e+08

Tableau 7: Comparaison du coût pour les différents algorithmes

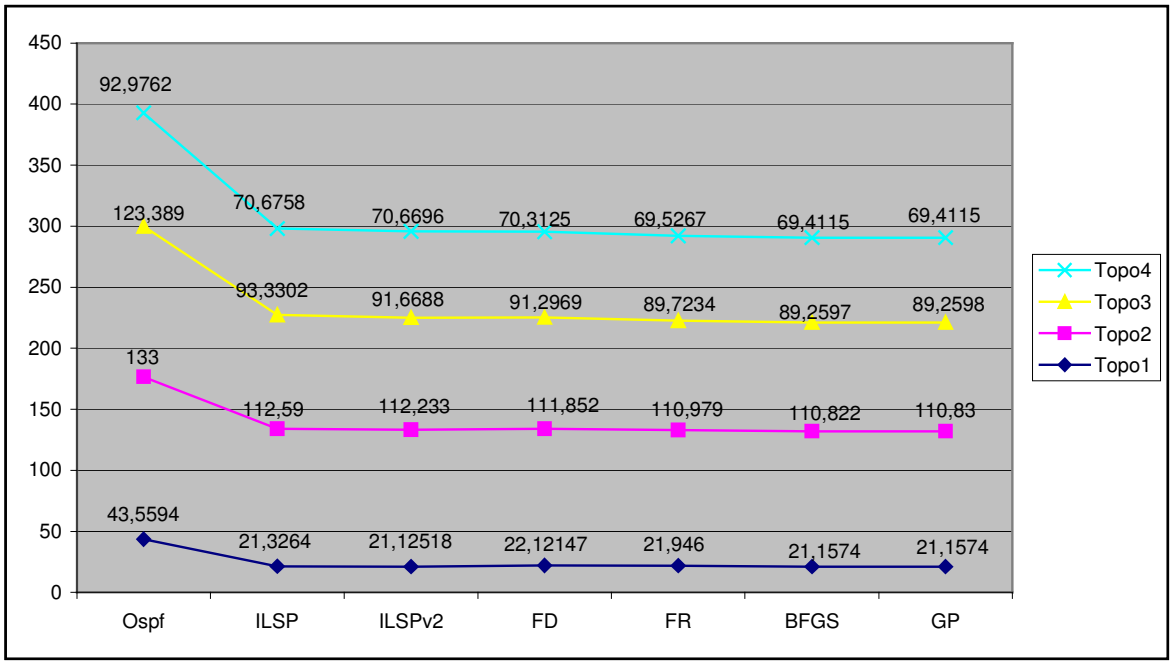


Figure 27: Comparaison du coût

Les résultats de ces tests montrent que les solutions obtenues par toutes les méthodes sont meilleurs que le routage OSPF de base. Donc encore une fois on gagne à utiliser un autre routage que celui proposé par défaut.

Les méthodes «BFGS» et «Gradient Projeté» obtiennent systématiquement de meilleurs coûts que les méthodes «Fletcher Reeves Projeté» et «Déviation de flots». ILSP et ILSPv2 s'approchent de manière intéressante de la solution optimale.

En générale nous observons le résultat suivant :

Coût(OSPf) > Coût(ILSP) > Coût (ILSPv2) > Coût(Fletcher Reeves) > Coût(Flow Deviation) > coût (Gradient Projeté) > coût (BFGS).

I.3. Comparaison du nombre d'itérations

#Itérations	Topo1	Topo2	Topo3	Topo4
ILSP	50	220	420	8320
ILSPv2	205	814	6342	5543460
Flow Deviation	34	12	1456	1436
Fletcher Reeves	449	87	872	464
BFGS	64	67	296	592
Gradient Projeté	64	76	365	592

Tableau 8: Comparaison du nombre d'itérations

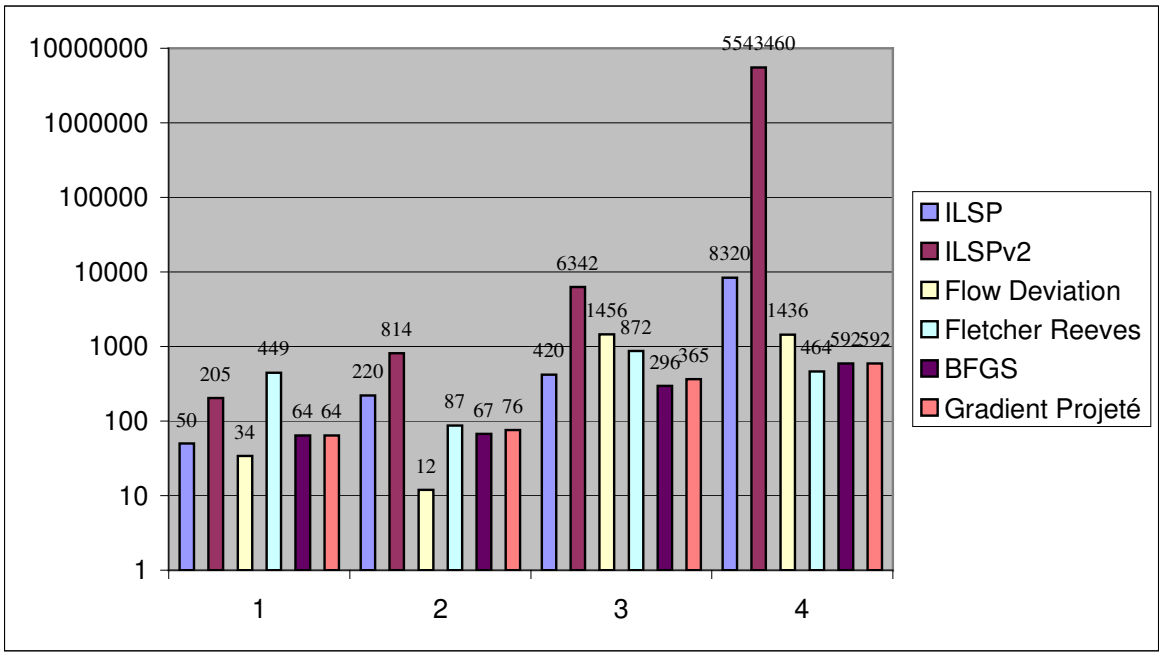


Figure 28: Comparaison du nombre d'itérations

Encore une fois les résultats obtenus sont ceux escomptés. Ainsi ILSPv2 est l'algorithme qui nécessite le plus d'itérations compte tenue de la décomposition en quantum qu'il doit utiliser pour router tout les LSPs.

« Déviation de flots » nécessite un grand nombre d'itérations. Cela est due à sa convergence très lente aux abords de l 'optimum (phénomène de zigzag).

« Fletcher Reeves projeté » en générale nécessite plus d'itérations avant convergence que « BFGS » ou encore « Gradient Projeté ».

I.4. Comparaison des temps d'exécution

Temps exécution (s)	Topo1	Topo2	Topo3	Topo4
ILSP	0,01	0,037	0,13	2,9
ILSPv2	0,15	0,49	1,95	4m25s
Flow Deviation	0,01	0,02	0,29	8,75
Fletcher Reeves	0,12	0,04	0,53	1,7
BFGS	2,59	3	90,1	4h36m25s
Gradient Projeté	0,01	0,018	0,14	1,8

Tableau 9: Comparaison du temps d'exécution

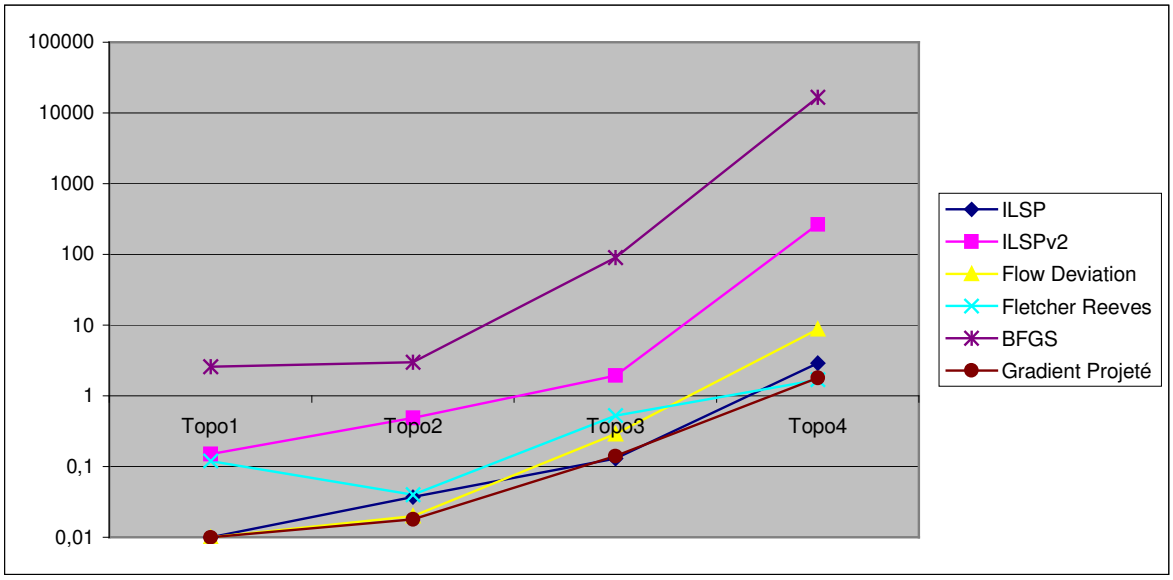


Figure 29: Comparaison du temps d'exécution

La hiérarchie précédente à presque été respectée mis à part le cas de BFGS. On peut confirmer que plus la taille du problème grandit plus l'assertion suivante est vérifiée

temps(BFGS) **>** temps(Flow Deviation) **>** temps(Fletcher Reeves) **>** temps(Gradient Projeté)

BFGS est un algorithme très puissant seulement pénalisé par son temps de calcul. En effet l'évaluation de la matrice H_k, à chaque itération, nécessite énormément de temps calcul plus la taille du problème est grande. dans le cas de la topo4 la taille de cette matrice est de (1223*1223) .

J. Conclusion

Dans ce chapitre nous avons comparé différentes méthodes non linéaires pour le problème de routage par partage de charge optimal. Nous avons opté après réflexion pour la méthode du gradient projeté. C'est une méthode qui fournit des résultats aussi bons que ceux de la méthode BFGS mais en beaucoup moins de temps.

Nous proposons une nouvelle méthode (ILSP) pour la recherche de chemins candidats pour le problème du mono-routage des LSPs. Cette méthode donne de bons résultats comparée aux méthodes non-linéaires présentées.

L'étude faite pendant le projet OPIUM a montré qu'ILSP génère les bon chemins candidats. En effet dans [DUH03] une version améliorée de l'algorithme « Flot Deviation » est initialisée par les chemins d'ILSP ou encore ILSPv2. Cette dernière génère rarement des chemins supplémentaires. Et quand cela se produit ils sont rarement retenus dans la solution finale.

ILSPv2 est une amélioration d'ILSP qui utilise un quantum unique pour tout les LSPs. Cette version à été développé surtout pour montrer que l'on peut approcher encore plus des solutions optimales obtenues par les algorithmes du type gradient. Et donc avoir un bon partage de charge sur les chemins générés au détriment d'un effort calculatoire plus important.

Ainsi la combinaison d'ILSP pour générer les chemins candidats et du gradient projetée pour obtenir le partage de charge optimal, est une bonne heuristique pour générer la solution multi-routée des LSPs.

« J'ai traversé ce pays de long en large et parlé aux personnes les plus érudites sur ce sujet, et je peux vous assurer que le traitement informatique des données n'est qu'une toquade qui ne passera pas l'année.»

L'éditeur en charge des publications techniques pour Prentice Hall, 1957.

Chapitre 4 - ETUDE DU ROUTAGE DES LSPS

A. Introduction

Le monde des télécommunications est en pleine évolution, et avec l'arrivée d'applications multimédia, l'opérateur de télécommunications se doit d'être garant d'une certaine qualité de service. Celle–ci passe non seulement par l'utilisation des dernières technologies développées à l'IETF telles que l'architecture Diffserv ou le protocole MPLS, mais aussi par l'optimisation de la planification du réseau : configuration des équipements, du routage et du dimensionnement des équipements.

La complexité des modèles de trafic (itératifs, non-linéaires…), les contraintes de QoS ou de sécurité et la complexité des réseaux (taille, contraintes technologiques…) font de chacun des problèmes précédents un problème d'optimisation extrêmement difficile qui ne peut être résolu qu'à l'aide d'outils logiciels puissants.

Plus précisément, les opérateurs ont besoin, entre autres, d'outils de planification qui leur permet de gérer le routage de milliers de LSPs. Avoir un outil qui permet d'optimiser les ressources réseaux tout en garantissant le respect des contrats de QoS établis avec les clients constitue un gain de temps et surtout d'argent. C'est dans cet esprit là, que l'algorithme d'optimisation des LSPs que nous présentons est développé.

Le problème que nous essayons de résoudre dans cette étude consiste comme énoncé en introduction à trouver un chemin unique (mono-routage) pour chaque LSP dans le cœur de réseau MPLS tout en tenant compte des différentes contraintes. Un problème similaire a été traité dans la littérature, routage de VP/VC dans les réseaux ATM et s'est avéré être de la classe NP-Complet en nombre de demandes [FRE99].

Plusieurs approches ont été développées telles que la programmation linéaire ([COS94], [ASH98], [CHA98], [CHA99]), les algorithmes de type glouton ([KIR83], [BON99]), les Meta-heuristiques (algorithmes génétiques ([COR99], réseaux de neurones [SCH97]) ou encore les techniques de satisfaction de contraintes (BIH : Blocking Island Hierarchy) [FRE99]. Toutes ces techniques ne traitent que le problème d'allocation de ressources dans les réseaux. De plus, aucune de ces approches ne permet à la fois de :

- ✓ Fournir au moins une solution faisable (s'il en existe une),
- ✓ Calculer une solution optimale par rapport à une métrique (fonction coût) ,
- ✓ Evaluer l'écart entre la solution fournie et une solution optimale (si celle si n'est pas atteinte),
- ✓ Quand le routage de la solution n'est pas possible (manque de bande passante) fournir les corrections à apporter au réseau,
- ✓ Fournir une solution après un temps de calcul très raisonnable,
- ✓ Trouver des routages de secours simultanément aux routages primaires,
- ✓ Trouver des routes optimisant des fonctions coûts (modèles de files d'attente) relatives à la qualité de service.

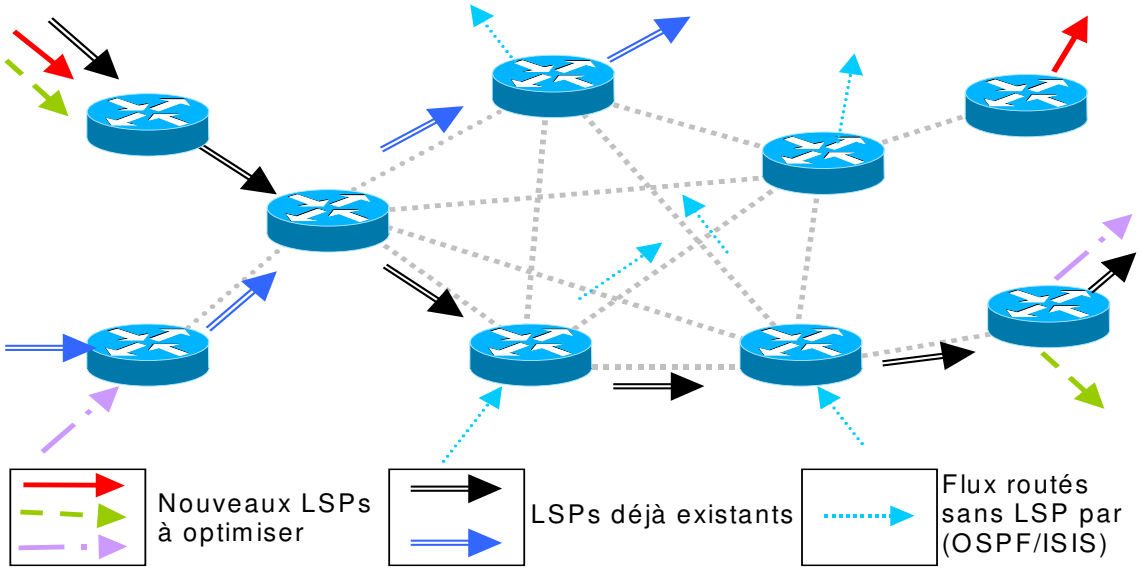


Figure 30: Cohabitation des LSPs avec les flux (OSPF/ISIS)

Dans cette étude nous proposons une heuristique qui répond à tous ces critères.

La programmation linéaire en {0,1} reste difficilement envisageable dès qu'il s'agit de résoudre des problèmes de tailles réelles (le nombre de variables devient astronomique). Il faut ajouter à cela que ce type de méthode ne fournit aucune indication dans le cas ou il n'existe pas de solutions admissibles. C'est pourquoi nous nous sommes dirigés vers une approche de programmation non linéaire qui combine gradient projeté (solution relaxée en partage de charge), le K-Routage et finalement les techniques de recherche aléatoire pour le difficile problème du mono-routage.

Plusieurs travaux ont été menés par l'équipe MRS du LAAS sur ce sujet. Durant un stage précédent j'ai eu l'occasion de travailler sur des méthodes de satisfaction de contraintes de type BIH [FRE99] et la méthode « Flow Deviation » [MIN74]. Cette expérience nous a conduit à développer une nouvelle approche à trois niveaux ILSP-OLS-ACO, qui est présentée dans les paragraphes suivants:

- **ILSP** (**I**terative **L**oading **S**hortest **P**ath),
- **OLS** (**O**ptimal **L**oad **S**haring),
- **ACO** (**A**nt **C**olony **O**ptimisation).

B. La méthode CISCO PCALC (Path CALCulation)

L'algorithme du calcul des routes de OSPF-TE est souvent appelé CSPF. Il s'agit du même type de calcul de plus court chemin qu'OSPF mais il permet d'optimiser les chemins les plus courts sur la base de métriques plus évoluées.

En fait, ce qui est présenté ci-après est la version de l'heuristique gloutonne proposée par CISCO (PCALC : Path CALCulation). On peut donc facilement

imaginer d'autres algorithmes, même heuristiques, basés sur les le calcul de plus courts chemins et exploitant simultanément des contraintes d'ingénierie de trafic.

Soit le réseau suivant :

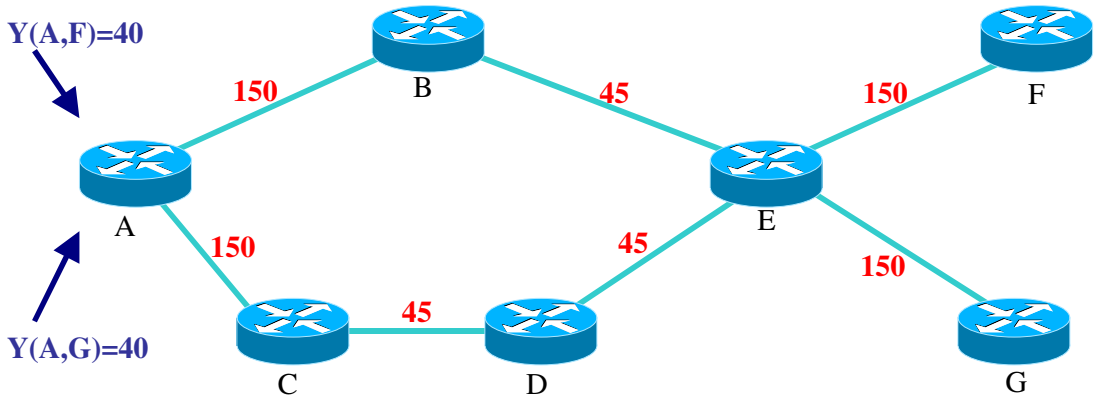


Figure 31: Le problème du poisson

Pour cet exemple toutes les interfaces ont une métrique de 1.

OSPF classique va trouver que le plus court chemin entre A et F est de coût 3 et est constitué des nœuds suivants :

A,B,E,F

De même OSPF classique va trouver que le plus court chemin entre A et G est de coût 3 et est constitué des nœuds suivants :

A,B,E,G

Les trafics Y(A,F) et Y(A,G) vont donc être routés sur une partie commune du chemin donné par ABE qui devra écouler 80 Mbit. Cependant, le lien BE ne peut écouler que 45 Mbit/sec. Ce qui donne 35 Mbit/sec de plus que la bande passante maximale du lien.

Une telle solution est donc inadmissible pourtant elle se produit régulièrement sur les réseaux dès que la charge en trafic augmente. En fait OSPF ne discerne pas les trafics Y(A,F) et Y(A,G) puisqu'il route uniquement en tenant compte de métriques préétablies et fixes.

CSPF introduit un calcul tenant compte de la métrique et d'une contrainte de bande passante, et, à la manière d'un algorithme glouton, va calculer progressivement ses chemins. CSPF va tout d'abord router le premier trafic, disons Y(A,F) en cherchant le plus court chemin jusqu'à F qui permet d'écouler 40 Mbit. Le chemin choisi sera donc A,B,E,F.

Cependant, après cette première affectation, CSPF va calculer les plus courts chemins en élaguant les liens du graphe qui ne disposent pas de suffisamment de bande passante résiduelle par rapport à la demande à placer. Il choisira donc le chemin A,C,D,E,G pour le deuxième trafic.

CISCO propose donc l'algorithme PCALC pour le routage des LSPs lorsque plusieurs chemins ont des bandes passantes admissibles.

L'algorithme se déroule comme suit :

- Tout d'abord trouver tous les chemins avec le coût de l'IGP (par exemple OSPF classique) le plus petit,
- Ensuite sélectionner les chemins ayant la plus grande bande passante sur tout le chemin. Cette bande passante sur le chemin est la plus petite bande passante des arcs constituant le chemin,
- Ensuite sélectionner le chemin avec le nombre de sauts minimum,
- S'il reste encore plusieurs chemins, en sélectionner un au hasard.

Nous allons en illustrer le fonctionnement sur l'exemple suivant:

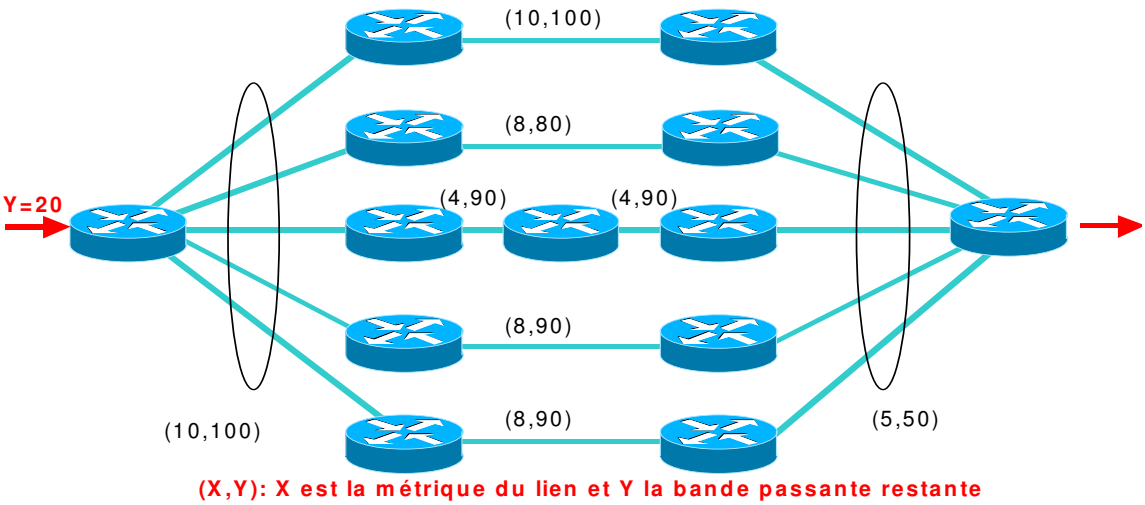


Figure 32: Ensemble des chemins possibles pour le LSP Y

Etape 1 : Le coût du chemin est 25, ce n'est pas le coût le plus faible : le chemin est éliminé.

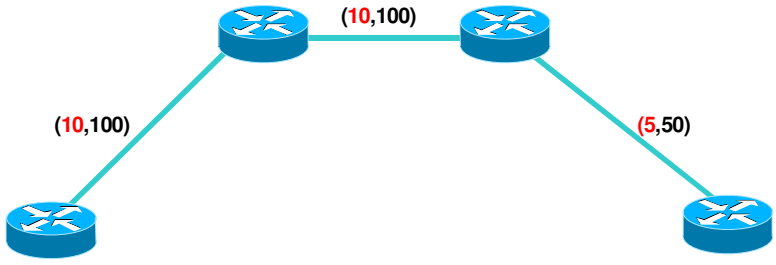


Figure 33: Elimination du chemin dont la métrique n'est pas minimale

Etape 2 : La bande passante est moins bonne que sur les chemins restants : Le chemin est éliminé.

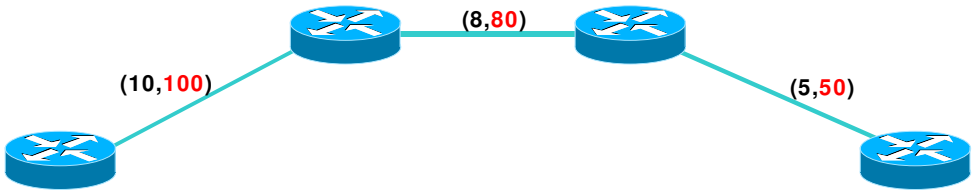


Figure 34: Elimination du chemin dont la bande passante est minimale

Etape 3 : La Bande passante du chemin est identique aux chemins restants, mais le nombre de sauts est supérieur : Le chemin est donc éliminé

Figure 35: Elimination du chemin dont le nombre de sauts est maximum

Etape 4 : Finalement le chemin optimal est choisi au hasard parmi les deux chemins restants

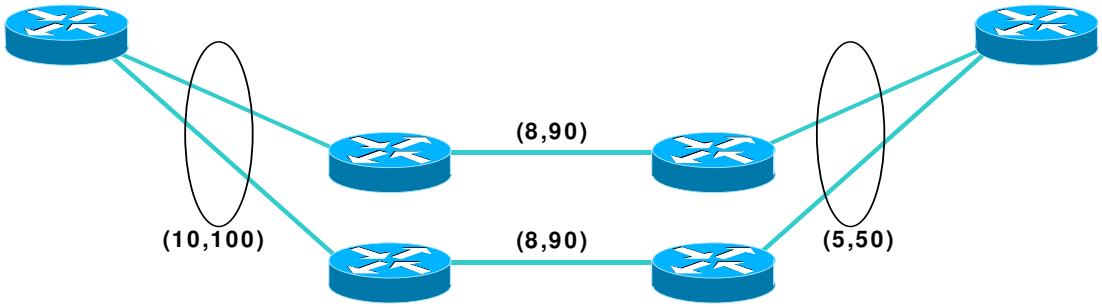


Figure 36: Choix aléatoire parmi les chemins restants

De part sa simplicité, PCALC est un algorithme « en ligne » implanté directement au niveau des routeurs du backbone MPLS. Les temps de calcul sont très petits car en plus il fonctionne à la demande.

A chaque fois qu'un LSP doit être routé le routeur « Ingress » fait appel a l'algorithme PCALC pour calculer un chemin. Il en résulte que cette approche ne possède qu'une vision locale du routage (mauvaise gestion des ressources du réseau).

Nous avons choisi d'implanter cet algorithme car c'est le plus répandu dans le monde des opérateur de télécommunication. Cela leur permet de prédire exactement les principaux indicateurs de leur réseaux (charge, délais, taux de pertes, …)

La version finale dont nous montrons les résultats en fin de chapitre est une version améliorée de PCALC qui tient compte de l'aspect sécurité et permet donc de calculer des routes de secours pour les LSPs protégés.

C. La Méthode ILSP-OLS-ACO

C.1. ILSP (Iterative Loading Shortest Path)

L'efficacité de tout algorithme de placement des LSPs aussi performant soit-il est pénalisée par le grand nombre de chemins possibles reliant les deux extrémités de la demande en LSP. L'idée principale d'ILSP est de réduire cette combinatoire en calculant un sous ensemble de chemins pour chaque LSP que l'on notera P_k pour (k=1…K).

La solution finale sera calculée sur la base de ces sous ensembles. La contrainte qui en découle c'est que ces sous ensemble doivent contenir la solution optimale. L'idée est donc de calculer les R meilleurs plus courts chemins (par rapport à la métrique bande passante ou encore QoS), avec R suffisamment grand.

Pour ce faire, ILSP charge progressivement le réseau avec des fractions de LSPs (d^k/R) en R itérations. A chaque itération j, un chemin candidat P_k^j est calculé pour chaque LSP k. au début de chaque itération j, nous définissons $\Omega_j = \{k \mid k = 1...K\}$ et pendant chaque itération j les opérations suivantes sont répétées K fois (pour tous les LSPs non marqués):

- ✓ Choisir un LSP $k \in \Omega_j$ selon une loi uniforme, router la demande (d^k/R) sur son plus court chemin $P_{j^*}^k$ au sens de la métrique $\Gamma(x)$ et d'affinité compatible et marquer le LSP,
- ✓ Faire $P_k := P_k \cup \{P_{j^*}^k\}$ et $\Omega_j = \Omega_j - \{k\}$,
- ✓ Mettre à jour la charge de toutes les interfaces appartenant à $P_{j^*}^k$ en propageant la fraction de LSP (d^k/R),
- ✓ Enlever la marque du LSP k.

ILSP permet de réduire considérablement la combinatoire initiale puisqu'elle fournit à la fin au plus R plus courts chemins par LSP (un plus court chemin pourra être découvert plusieurs fois).

Trier les demandes en LSP par ordre décroissant de priorité et de bande passante (pour la même priorité) permet d'améliorer la solution finale dans le sens ou les ressources du réseau sont d'abord offertes aux LSPs les plus prioritaires et les plus gourmands en ressources. On calcule pour ces LSPs de meilleurs plus courts chemins.

Le quantum routé lors des itérations d'ILSP a aussi un impact prépondérant sur la qualité des PCC générés pour chaque flot. Notamment lorsque les demandes de ceux ci ne sont pas du même ordre de grandeur (Ko et Mo). En effet les « grosses

demandes » ont tendance à écraser les petites dans la mesure ou elles s'accaparent la quasi totalité des « bons chemins ».

Une idée d'amélioration pour ILSP est d'avoir un seul quantum pour tous les flots. Cela permet d'explorer plus de routes pour tous les flots et donc d'améliorer nécessairement la solution finale. Seulement l'inconvénient de cette approche provient du nombre d'itérations lorsque les demandes ne sont pas du même ordre.

D'autres méthodes de générations de chemins existent dans la littérature. Les plus connues sont la méthode « K-Shortest-Path », la méthode de « Flow-Deviation » ou encore la méthode de « Génération de Colonnes ». Pour la première elle nous à semblé inadaptée car elle se base sur des métriques fixes qui ne dépendent pas du trafic. Pour les suivantes l'idée défendue est que ILSP compte tenu des comparaisons faites dans le chapitre 3 est un algorithme suffisamment performant pour ne pas rater les « bons chemins » ce qui combiné à sa rapidité permet de dire que c'est un bon compromis pour la génération des chemins admissibles.

Finalement ILSP permet aussi de gérer la sécurisation des LSPs car pour un LSP protégé, elle ne garde dans les plus courts chemins retenus que ceux qui possèdent au moins un chemin nœuds-disjoints. Nous reviendrons sur ce point plus en détails dans le chapitre 5.

C.2. OLS (Optimal Load Sharing)

OLS résout le problème du routage optimal en partage de charge. OLS se base essentiellement sur les informations données par le gradient de la fonction multicritères citée auparavant.

Le problème peut être formulé de la façon suivante :

$$\begin{cases} \quad Minimiser \;\; \Gamma(x) \\ \quad\quad sous \\ \sum_{j=1}^{p(K)} x_j^k = 1, k = 1...K \\ x_j^k \geq 0,\; j = 1...p(K), k = 1...K \end{cases} \tag{4.1}$$

avec *p(K)* = $|P_k|$ calculé par ILSP et ou $\Gamma(x)$ est l'une des fonctions multicritères définies précédemment (1.5), (2.6) ou encore (2.11).

Le problème non linéaire (4.1) peut être résolu par des méthodes se basant sur les techniques de directions de descentes. Nous avons choisi de le résoudre par la méthode du gradient projeté (très efficace même au voisinage du point optimum).

Pour ce problème les performances du gradient projeté sont très bonnes, compte tenu du fait que l'ensemble des routes à été considérablement réduit par ILSP (4 à 7 routes par LSP en général).

L'itération *q* de cet algorithme fonctionne de la façon suivante :

$$x_j^k(q+1) = x_j^k(q) + \nu \cdot \left\{ (p(k)-1) \cdot \frac{\partial \Gamma}{\partial x_j^k} - \sum_{i \neq j} \frac{\partial \Gamma}{\partial x_i^k} \right\} \quad (4.2)$$

C'est la projection du gradient sur l'ensemble des routes admissibles (ensemble construit à l'étape précédente grâce à ILSP), cette projection tient compte de la contrainte d'intégrité du flot ($\sum_{j=0}^{p(k)} x_j^k = 1, \ \forall k = 1...K$).

Le gradient projeté permet de déterminer de façon très efficace le routage optimal en partage de charge $\overline{x}_j^k$ (optimal par rapport au sous-ensemble des routes retenues par ILSP). Cependant, à la convergence, et au même titre que la méthode « Déviation de flots », la solution optimale à partage de charge n'est pas la solution optimale du problème initial de mono-routage.

Nous soulignons que nous n'avons pas besoin de s'approcher très prés de l'optimum optimorum car nous le rappelons cette étape permet d'affiner les répartitions des LSP sur les chemins disponibles et que cela permet d'initialiser l'algorithme des fourmis de la phase suivante.

La phase OLS présente deux avantages majeurs:

- ✓ Elle réduit encore plus la combinatoire pour ACO car d'un coté elle élimine toutes les routes telle que $\overline{x}_j^k = 0$, et elle « mono-route » d'un autre coté une bonne partie des LSPs ($\overline{x}_j^k = 1$).
- ✓ Elle permet de fournir une bonne solution initiale pour ACO (distribution des probabilités initiales).

Au bout d'un certain nombre d'itérations, on est confronté à deux possibilités :

- Solution mono-routée : pour chaque LSP une route unique a été retenue, ce qui implique la fin du processus d'optimisation.
- Solution multi-routée : quelques LSPs (voir tous) possèdent encore plusieurs routes candidates. Dans ce cas le processus est complété par la troisième phase (ACO).

C.3. ACO (Ant Colony Optimization)

C.3.1. Introduction

L'optimisation par colonies de fourmis est une technique inspirée par un travail de biologistes [DEN83] repris par des informaticiens [MOY88] et largement exploité et développé par Marco Dorigo dans les années 90 ([DOR99], [BON99]).

L'idée consiste à imiter le comportement des fourmis réelles qui collaborent, par exemple pour la recherche de sources de nourriture en mélangeant comportement

d'exploration aléatoire et suivi des traces chimiques laissées par leurs congénères. Ces traces chimiques, les « phéromones », sont utilisées par les fourmis pour communiquer entre elles de manière indirecte.

Chaque fourmi se dirige en tenant compte des traces de phéromone qui sont déposées par les autres membres de la colonie qui la précèdent. Comme cette phéromone s'évapore, ce choix probabiliste évolue continuellement.

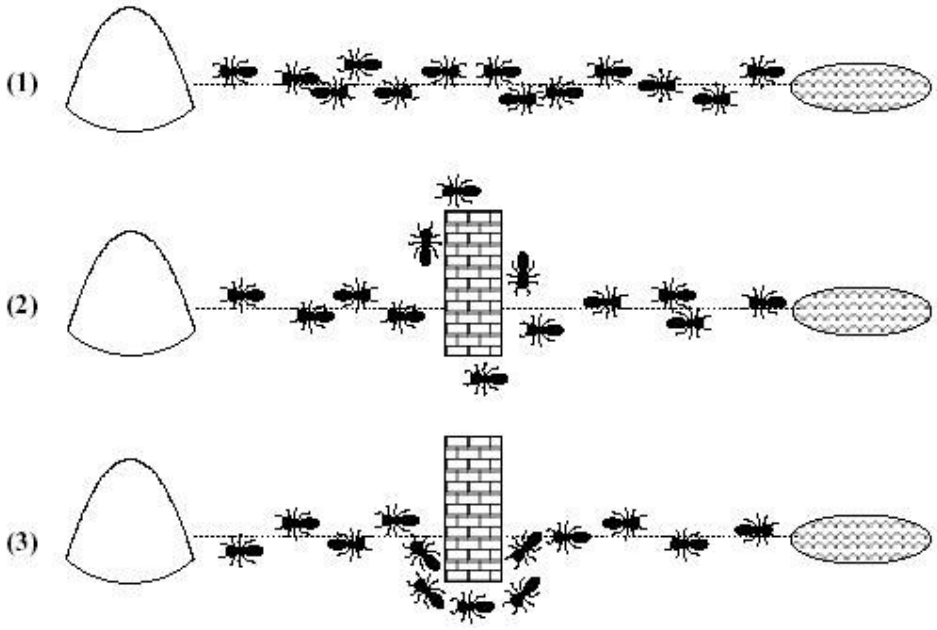


Figure 37: Rôle de la phéromone sur le choix du chemin par les fourmis

1. Des fourmis réelles suivent un chemin entre le nid et une source de nourriture,
2. Un obstacle survient sur le chemin, les fourmis choisissent de tourner à gauche ou à droite, avec des probabilités égales; la phéromone est déposée plus rapidement sur le chemin le plus court,
3. Toutes les fourmis ont choisies le chemin le plus court.

Ce comportement collectif, basé sur une sorte de mémoire partagée entre tous les individus de la colonie, peut être adapté et utilisé pour la résolution de problèmes d'optimisation combinatoire ou de satisfaction de contraintes.

D'abord, appliquée au problème du voyageur de commerce, l'optimisation par colonies de fourmis a rapidement prouvé son efficacité dans le cadre de l'optimisation combinatoire en général est s'est montrée particulièrement profitable pour le problème du routage des paquets d'information dans les grands réseaux d'interconnexion. C'est dans ce cadre là que nous allons utiliser cet algorithme.

C.3.2. Adaptation d'ACO au problème du mono-routage

Dans une itération d'algorithme ACO, f agents (fourmis) construisent chacun une solution d'après des décisions basées sur des critères heuristiques et sur des traces

de phéromone. Les traces sont mises à jour en examinant les solutions obtenues. Elles sont renforcées pour les décisions ayant donné de meilleures solutions et diminuées pour les autres.

Ce mécanisme permet d'améliorer progressivement les solutions au cours des itérations. En pratique, on construit *f* solutions initiales, puis on répète l'itération générale suivante jusqu'à la réalisation d'un critère d'arrêt comme un nombre maximum d'itérations ou un écart donné par rapport à une borne inférieure :

- Mise à jour des traces de phéromone dans le réseau,
- Génération de f nouvelles solutions par les fourmis, en exploitant les traces de phéromone,
- Application avec une probabilité donnée d'une recherche locale à ces solutions.

Ainsi dans notre modélisation ACO permet de fournir une solution mono-routée à partir de la solution partielle de OLS. L'enchaînement des deux algorithmes permet non seulement d'initialiser les probabilités initiales pour ACO mais elle permet aussi de laisser de côté tous les LSPs mono-routés pas OLS. Ceux là ne sont plus modifiés par ACO.

Dans notre algorithme ACO, les fourmis construisent récursivement des solutions pour le placement des LSPs par une exploration aléatoire de l'ensemble des solutions possibles. La distribution de probabilités initiale provient des $\overline{x}_j^k$ calculés par le gradient projeté à l'étape précédente.

Durant chaque itération *t*, chaque fourmi *l =1..m* construit une solution en *K* étapes en routant LSP par LSP. Chaque fourmi commence par un LSP choisi de façon aléatoire et choisit le LSP suivant selon une loi probabiliste de transition.

Plus précisément, durant l'itération *t* la fourmi *l* itère les étapes suivantes jusqu'à ce que tous les LSPs soient routés.

Soit $\Omega_t^l = \{k \mid k = 1...K\}$ l'ensemble des LSPs à router pour la fourmi *l* à l'itération *t*

- Choisir de façon aléatoire le LSP $k \in \Omega_t^l$ selon une loi uniforme,
- Choix aléatoire de la route pour le LSP *k* selon la probabilité:

$$P_l^k(j,t) = \frac{\left[\tau_j^k(t)\right]^\alpha \cdot \left[\eta_j^k(t)\right]^\beta}{\sum_{i=1}^{p(k)} \left[\tau_i^k(t)\right]^\alpha \cdot \left[\eta_i^k(t)\right]^\beta} \tag{4.3}$$

$P_l^k(j,t)$ est la probabilité que la fourmi *l* route le LSP *k* sur son chemin *j* à l'itération *t* ;

$$\eta_{\mathrm{j}}^{\mathrm{k}} = \frac{1}{\sum_{\mathrm{ue}\,\mathrm{P}_{\mathrm{j}}^{\mathrm{k}}} \Gamma(\mathrm{y_u} + d^k, C_u)} \tag{4.4}$$

η_j^k est l'augmentation du coût due au routage du LSP *k* sur le chemin P_j^k ;

τ_j^k est la trace (virtuelle) de phéromone associée au routage du LSP k sur le chemin P_j^k, cette trace est mise à jour à la fin de chaque itération *t*. Elle sert à conditionner les décisions de routage futures des fourmis (plus cette quantité est grande plus le chemin a des chance d'être re-sélectionné) ;

α et β sont deux paramètres ajustables qui permettent de contrôler les poids relatifs τ_j^k (intensité de phéromones) et η_j^k (visibilité) ;

- Mettre à jour les charges des interfaces en propageant la demande d^k sur le chemin $P_{j^*}^{k,l}(t)$.

Toutes les fourmis répètent *K* fois les étapes précédentes pour construire une solution. A la fin des *K* itérations, la trace de phéromones associée aux décisions de routage est mise à jour de la façon suivante :

$$\tau_j^k(t+1) = (1-\rho)\cdot\tau_j^k(t) + \sum_{i=1}^{m} \Delta\tau_j^{k,l}(t) \tag{4.5}$$

ρ est le coefficient de diminution de l'intensité de la trace de phéromones, il permet d'assurer une exploration efficace du domaine des solutions.
$\Delta\tau_j^{k,l}(t)$ est la quantité de phéromones que la fourmi *l* associe au routage du LSP *k* sur le chemin P_j^k, cela dépend du coût de la solution de la fourmi *l* .

$$\Delta\tau_j^{k,l}(t) = \begin{cases} Q/\Gamma(\bar{x}) & si\ P_{j^*}^{k,l}(t) = P_j^k \\ 0 & \text{sinon} \end{cases} \tag{4.6}$$

Le dénominateur ($\Gamma(\bar{x})$) étant le coût de la solution de la fourmi *l* et Q un paramètre (d'après [BON99], ce paramètre influence très peu la solution finale, sa valeur par défaut vaut le coût de la solution optimale en partage de charge).

Initialement la trace de phéromones $\tau_j^k(0)$ vaut ($\tau_0 \cdot \bar{x}_j^k$), où τ_0 est une petite valeur positive et $\bar{x}_j^k$ est la fraction du LSP *k* routée sur le chemin P_j^k par la solution optimale en partage de charge.

Le fonctionnement d'ACO permet d'obtenir une solution après une itération. L'utilisateur peut donc décider d'améliorer sa solution en augmentant le nombre d'itérations (interactivité), jusqu'à ce qu'il obtienne un coût satisfaisant.

A la fin d'ACO, soit nous avons une solution complète (tous les LSPs ont été placés) soit certains LSPs n'ont pas pu être placés par manque de bande passante. Pour chacun de ces LSPs non seulement on fournit le meilleur chemin que l'on a calculé, mais en plus nous fournissons des indications de redimensionnement, pour les liens et les interfaces, afin que ce chemin soit admissible.

D. Tests et résultats

D.1. Approche bande passante

Dans cette section nous exposons les résultats obtenus par la méthode ILSP-OLS-ACO. Le critère optimisé est la bande passante résiduelle totale du réseau.. Ces résultats sont comparés à ceux obtenus par PCALC. Pour l'algorithme PCALC les LSPs ont étés triés par ordre décroissant de priorités et par bande passante requise décroissante au sein de la même priorité.

Pour ce faire nous allons utiliser l'ensemble de réseaux tests suivant

	#Noeuds	*#Liens*	*#LSPs*	*#Variables*	*#Solutions*
Test1	**5**	**6**	**5**	**7**	**4**
Test2	**15**	**21**	**49**	**102**	**$\approx 10^{12}$**
Test3	**15**	**22**	**49**	**93**	**$\approx 10^{12}$**
Test4	**30**	**75**	**92**	**383**	**$\approx 10^{50}$**
Test5	**30**	**75**	**435**	**1154**	**$\approx 10^{151}$**
Test6	**30**	**75**	**947**	**3048**	**$> 10^{333}$**

Tableau 10: Caractéristiques des topologies de test

D.1.1. Comparaison des coûts

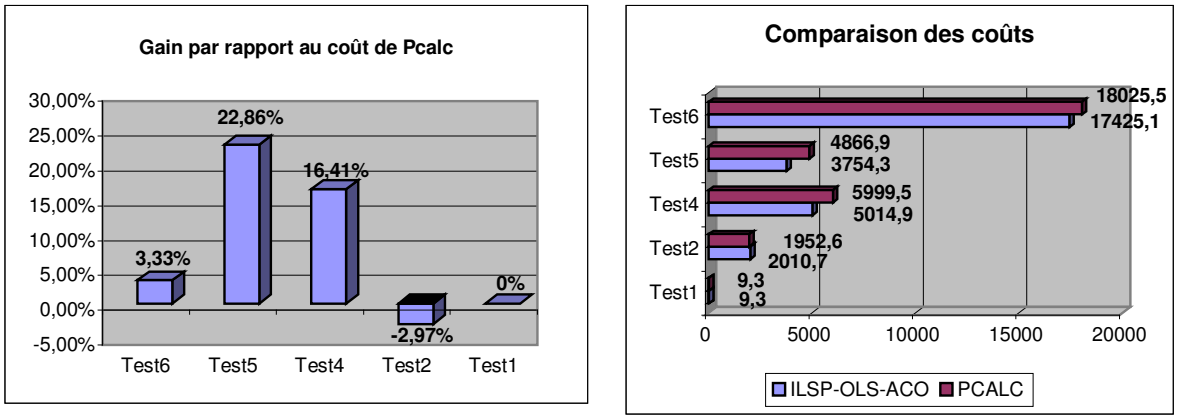


Figure 38: Comparaison des coûts

Dans le cas de réseau contraint où les bandes passantes utilisables sont taillées raisonnablement voir au plus juste, l'allocation de la moindre ressource conditionne la solution finale. La vision locale due à l'aspect glouton de Pcalc ne lui permet pas de garantir une solution en présence de tels réseaux.

A l'encontre de Pcalc ILSP-OLS-ACO possède une vision globale du schéma de routage et peut donc garantir une solution à chaque fois qu'il en existe une.

Le premier constat que l'on peut établir concerne le coût de la solution finale. En effet la qualité de la solution trouvée par ILSP-OLS-ACO permet non seulement d'obtenir un meilleur coût par rapport à Pcalc. Mais elle permet aussi d'obtenir des gains conséquent par rapport à une solution de type Pcalc.

Le cas du test2 n'est pas une anomalie dans la mesure ou Pcalc n'as pas placé 3 LSPs donc il est normale que ILSP-OLS-ACO consomme plus de bande passante et obtienne un coût supérieur.

D.1.2. Comparaison des bandes passantes résiduelles

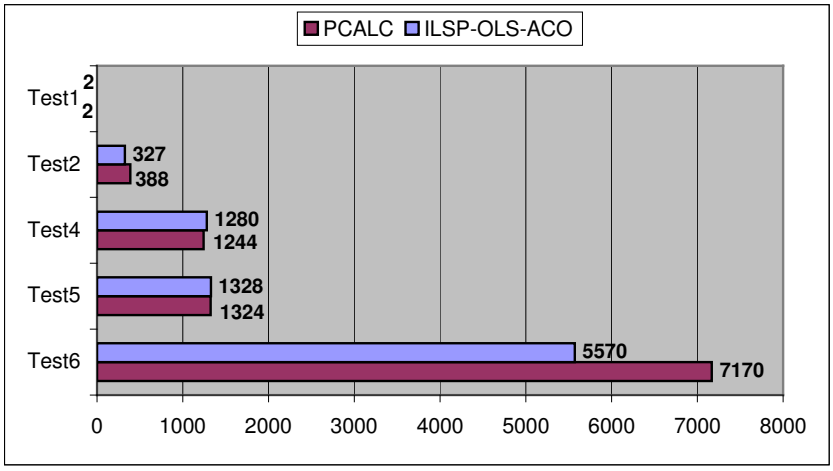


Figure 39: Comparaison de la bande passante résiduelle

Le deuxième constat qui peut être établi est le fait que ILSP-OLS-ACO dégage dans ses solutions plus de bande passante résiduelle que PCalc. Ce qui est crucial pour un opérateur dans la mesure ou celle-ci peut être utilisée pour le placement des futures demandes ou encore en cas de pannes.

D.1.3. Comparaison du nombre de LSPs placés

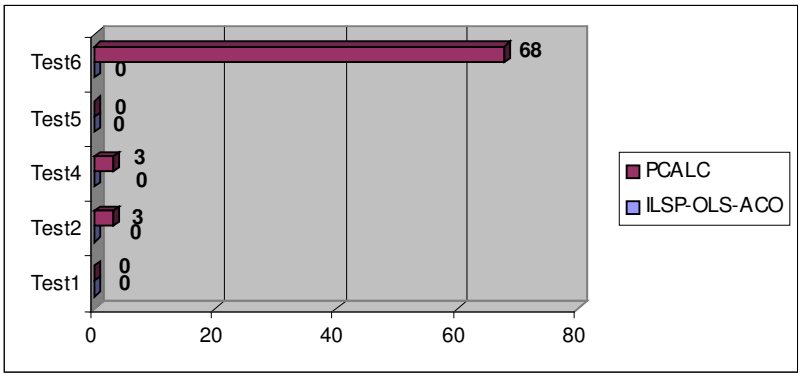


Figure 40: Comparaison du nombre de LSPs placés

Le troisième constat illustre la capacité d'ILSP-OLS-ACO à trouver systématiquement une solution (lorsque celle-ci existe) alors que Pcalc n'arrive pas à placer tous les LSPs.

D.1.4. Comparaison des temps d'exécution

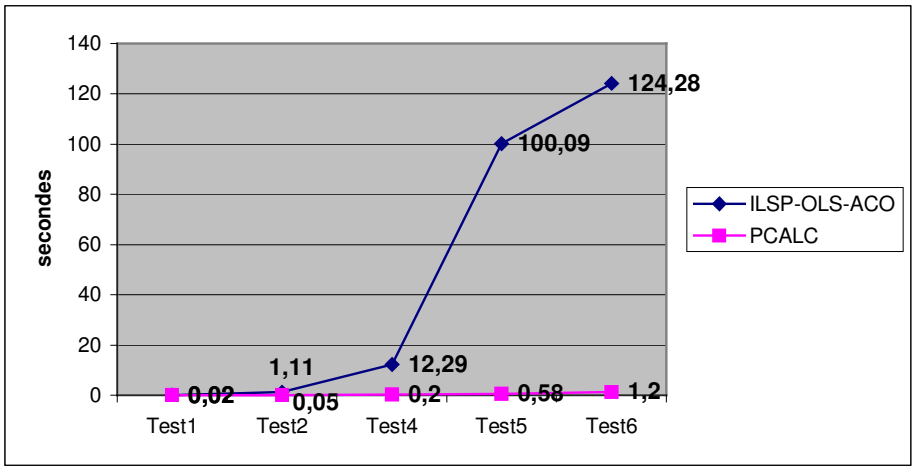


Figure 41: Comparaison des temps d'exécution

La comparaison des deux méthodes, sur ces réseaux tests, nous amène à plusieurs conclusions. La première conclusion concerne les temps d'exécution. PCALC est un algorithme très rapide. Cependant cela reste le seul avantage majeur par rapport à la méthode ILSP-OLS-ACO.

En effet, PCALC ne permet pas de placer tous les LSPs (Test6,Test2,Test4) alors que ILSP-OLS-ACO y parvient. ILSP-OLS-ACO fourni des solutions de moindre coût par rapport à PCALC et permet aussi de dégager plus de bande passante résiduelle (utilisable pour de futurs placements de LSPs).

D'autre part les temps d'exécution de l'algorithme ILSP-OLS-ACO restent tout à fait correctes compte tenu de la taille des problèmes traités.

D.2. Approche QoS

L'algorithme ILSP-OLS-ACO est plus performant que PCALC quand le critère optimisé est la bande passante résiduelle totale du réseau. Cependant il n'y a aucune garantie sur la satisfaction des contraintes de QoS. C'est pourquoi nous allons montrer dans ce qui suit qu'il est important de prendre en compte ce type de critères.

Nous allons comparer la solution en prenant comme fonction coût d'un côté celle définie par (1.5) et de l'autre côté celle définie par (2.11). Nous utilisons dans un premier temps le modèle PQ/WFQ. Nous montrons par la suite quelques résultats obtenus par le modèle M/M/1/N.

La comparaison se fera suivant trois critères importants : le nombre de LSPs hors QoS, l'écart moyen globale par rapport au contraintes de QoS et enfin la bande passante résiduelle totale du réseau.

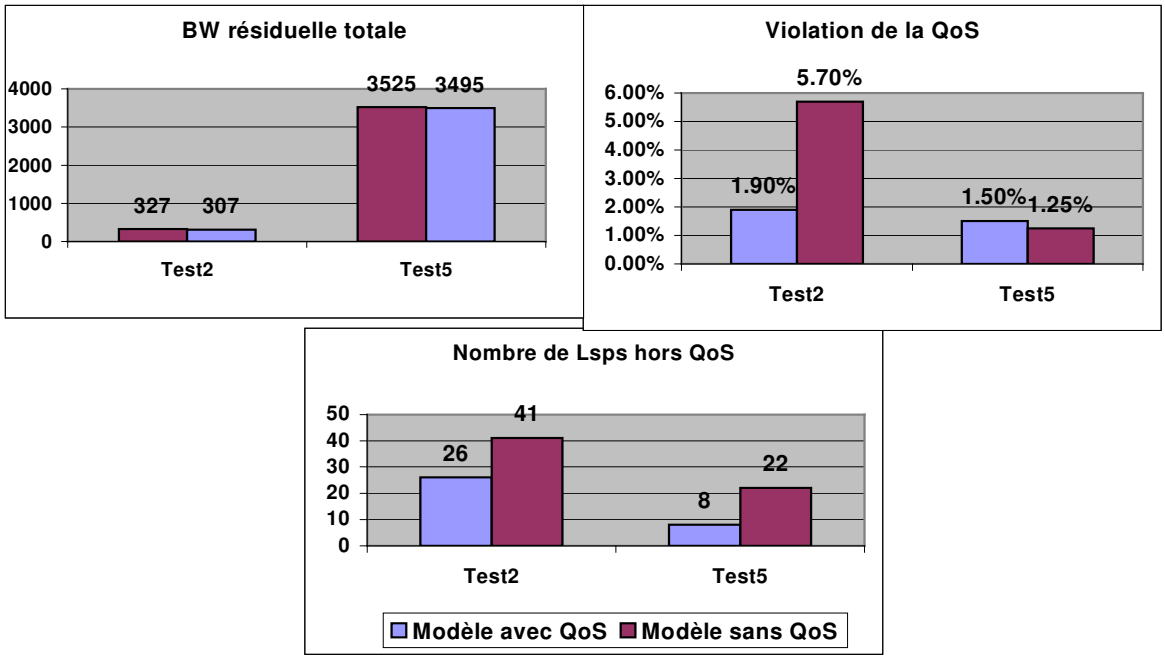


Figure 42: Comparaison des modèles avec et sans QoS

Cette comparaison va dans le sens prévu. Ajouter un modèle d'évaluation de performance au processus d'optimisation permet une meilleur gestion de la QoS. Comme on peut le voir sur les graphiques le modèle avec QoS permet non seulement de réduire le nombre de LSPs, pour lesquels les contraintes de QoS ne sont pas satisfaites, mais en plus réduit leur écart moyen. Pour le Test2 l'écart par rapport au délai maximum toléré passe de 5.70% à 1.90%.

Dans le cas du test5 le fait de ramener 14 LSPs dans la zone d'acceptabilité des contraintes de QoS dégrade un peu la violation de ceux qui restent en dehors de cette zone.

Dans les deux cas, cette solution consomme un peu plus de bande passante car elle déroute certains LSPs de leurs PCC pour les faire passer par des chemins un peu plus long. Cela implique d'un côté la réduction de la charge de certaines interfaces qui vont induire moins de délai et de pertes. De l'autre côté cela augmente la consommation en bande passante car le LSP occupe plus de ressources.

Il est à noter que ces deux tests sont très contraints, dans d'autres cas ce modèle permet de placer tous les LSPs (respect des contraintes de capacités) et de garantir leur contraintes de QoS.

Nous avons aussi comparé les performances du modèle M/M/1/N à celles du modèle PQ/WFQ sur le réseau test3. nous avons obtenu les résultats suivants :

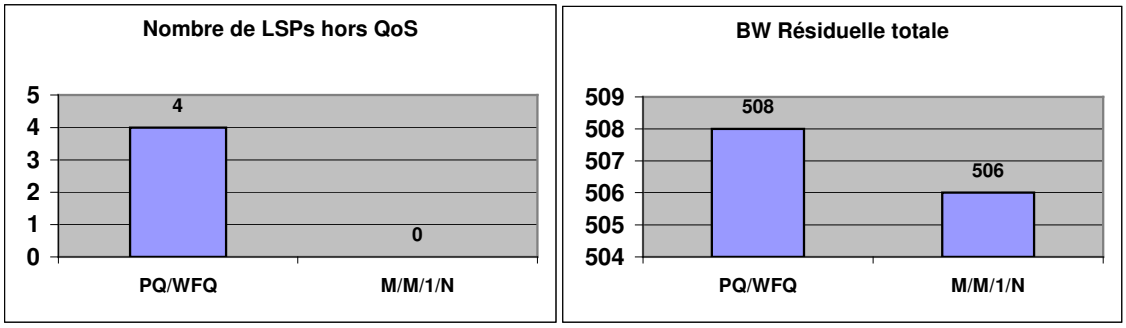
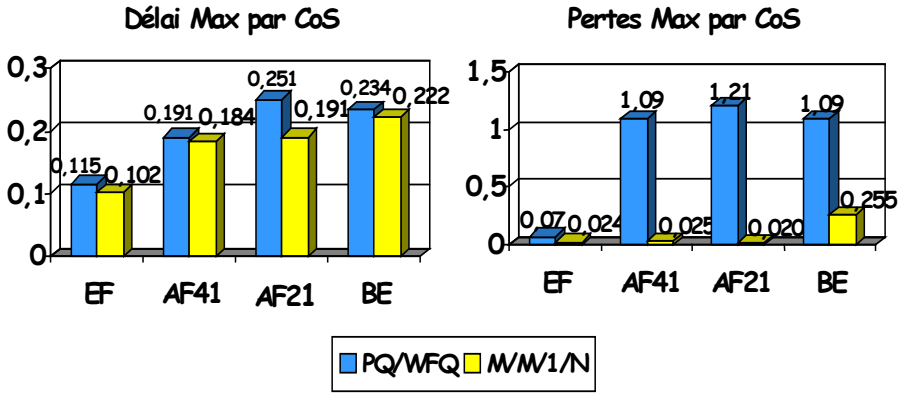


Figure 43: Comparaison des deux modèles

On peut constater dans ce cas que le modèle M/M/1/N trouve une solution pour les 4 LSPs qui satisfait leurs contraintes de QoS.

Nous avons observé ce qui se passe aussi au niveau des délais et pertes par classes. Nous présentons ces résultats dans les graphiques suivants :



Comme on peut le constater, le modèle M/M/1/N permet d'obtenir une solution dont les délais maximum par classes sont meilleurs que ceux obtenus par le modèle PQ/WFQ. Il en va de même pour les pertes maximum par classe. Nous attribuons cet différence au calibrage de la fonction coût (2.11) adoptée pour le modèle PQ/WFQ. En effet il arrive que la partie du coût due au critère bande passante affecte plus le coût total. Ce coût influence donc plus la solution finale dans le sens ou il favorise une solution satisfaisant les contraintes de capacité au détriment des contraintes de QoS.

E. Conclusion

Dans ce chapitre nous présentons une nouvelle méthode de résolution du problème de mono-routage des LSPs. Cette méthode comme nous l'avons décrite permet de tenir compte d'un nombre important de contraintes. Nous l'avons conçue au plus près des exigences des opérateurs. En dépit de cela, elle permet de gérer des réseaux de tailles réelles combinés à un grand nombre de LSPs, en des temps plus que raisonnables.

Nous avons effectué une comparaison avec l'algorithme Pcalc de CISCO afin de justifier l'approche « hors ligne » du mono-routage des LSPs. En effet en utilisant ILSP-OLS-ACO les opérateurs on plus de garanties quand à la qualité de la solution au détriment d'un temps de calcul à peine plus important.

La comparaison entre l'approche bande passante et l'approche QoS permet de justifier un modèle plus complexe et donc plus « lourd » à gérer.

Nous montrons par les tests que nous avons effectués que l'on gagne à utiliser ce modèle en présence d'applications sensibles à la QoS. Ce modèle permet non seulement de trouver un routage qui satisfait les contraintes de chacune des applications (quand cela est possible). Mais il permet aussi de réduire la violation des contraintes des LSPs pour lesquels il n'a pas pu trouver de chemins qui satisfont les contraintes de QoS.

« Mais à quoi cela peut-il bien servir ?? »

Un ingénieur d'IBM commentant l'invention de la puce électronique, 1968.

Chapitre 5 - SECURISATION DES RESEAUX MPLS

A. INTRODUCTION

L'utilisation d'IP-MPLS pour le transport de services temps réel impose une excellente disponibilité du réseau. De tels services imposent notamment de pouvoir garantir un rétablissement de la connectivité en moins de 50 ms, en cas de panne de lien ou de nœud IP-MPLS.

Les méthodes actuelles de protection SDH (AIS, MS-SPRING) permettent de garantir ces temps de sécurisation. En revanche, elle sont très coûteuses en ressources car elle nécessitent de dédier des liens à la protection, et ne permettent de protéger le trafic que contre les pannes de liens, et non contre les pannes de routeur. Il est donc préférable de réaliser la sécurisation des liens et des routeurs directement au niveau de la couche IP-MPLS.

Les pannes des éléments du réseau peuvent être d'origines diverses. Ainsi les pannes de liens peuvent être la cause d'erreurs sur des chantiers de travaux publiques (coups de pelleteuse sur une fibre optique) ou plus simplement d'un lien débranché. En ce qui concerne les routeurs cela peut provenir d'une panne de courant. A cela il faut ajouter les pannes logicielles dues à des erreurs humaines.

Tous les éléments d'un réseau sont susceptibles de tomber en panne. Pour garantir un haut niveau de disponibilité du réseau, il faut donc prévoir ces pannes et déterminer des méthodes automatiques pour les détecter et assurer la continuité du service le plus rapidement possible.

B. Les mécanismes de protection

Plusieurs approches existent dans la littérature pour protéger les réseaux IP-MPLS contre les pannes de liens et de routeurs. Ces mécanismes de protection suivent un cycle, qui commence quand la faute est détectée et qui finit quand le LSP est recouvert. Ce cycle doit comporter une méthode pour choisir les deux chemins, principal et de protection, et une méthode pour réserver la bande passante pour ces chemins.

Des mécanismes de détection de pannes le long du chemin et de notification de pannes sont encore nécessaires pour envoyer l'information aux entités responsables de réagir aux pannes et de prendre les actions correctives. Finalement, la méthode utilisée doit comporter un mécanisme de commutation du chemin principal vers le chemin de protection.

Parmi les mécanismes de protection, nous allons décrire le modèle de réparation globale (Backup), le modèle de réparation locale ou restauration d'un segment d'un LSP (Fast Reroute), et enfin la protection à plusieurs niveaux (Multi-Layer).

Dans le premier modèle, un nœud d'entrée est responsable d'effectuer la restauration suite à la réception du signal d'indication de panne quelque soit le lieu où la faute est intervenue le long du chemin principal. Cette méthode nécessite un chemin de Backup disjoint pour chaque chemin principal.

Dans le cas d'une réparation locale, la protection a lieu sur une partie du chemin principal et la procédure de restauration commence tout simplement par le point de défaillance. Le dernier modèle de protection à différents niveaux est applicable dans le cas de scénarii avec des pannes multiples.

B.1. La protection de Chemin (Backup)

Le protocole MPLS permet d'établir et de maintenir automatiquement les LSPs à travers le cœur de réseau en utilisant le protocole de réservation de ressource (RSVP). Le chemin employé par un LSP dépend étroitement des ressources disponibles dans le réseau.

Pour des services fondés sur des critères de qualité de service, on peut escompter obtenir le niveau de qualité garanti en sélectionnant des routes concordant avec ces critères. MPLS est aussi employé par les opérateurs pour améliorer la tolérance aux pannes du réseau lorsqu'un incident intervient sur un nœud de réseau ou un lien. La protection des liens repose sur l'établissement d'un chemin de secours entre le routeur d'entrée et le routeur de sortie pour chaque LSP primaire créé.

La panne du LSP principal provoque le basculement du trafic sur le LSP de secours, pré configuré ou établi dynamiquement. Le chemin du LSP est calculé à la source de celui ci. En cas de pannes la source du LSP détermine un nouvel itinéraire pour le LSP. Le calcul de ce chemin de Backup prévoit une utilisation optimale des ressources.

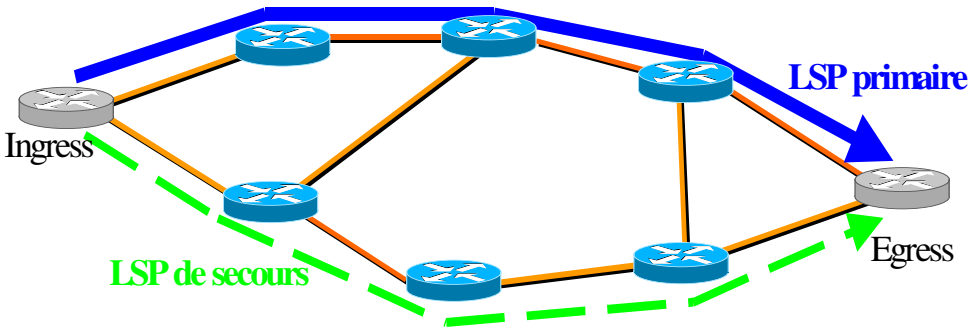


Figure 44: Protection du LSP par chemin de Backup

Le LSP de Backup est le mode de protection par défaut de MPLS-TE. Il faut distinguer deux sous modes pour ce type de protection :

- Le mode « ***Head-End Backup*** » ou le Backup n'est ni calculé ni signalé donc tout se fait après panne et durant ce laps de temps dû aux délais de

transmission des messages, un nombre important de paquets vont être perdus.

- Le mode « ***Stand-by LSP*** » est une extension de RSVP-TE où le Backup est signalé et prêt à l'emploi l'inconvénient dans ce cas est la sur-réservation de bande passante.

Un LSP de secours pré-alloué peut être utilisé pour protéger plusieurs LSPs primaires, ce qui permet d'économiser des ressources. Dans ce cas, on considére que des pannes simultanées affectant plusieurs LSPs partageant un même LSP de secours sont très peu probables.

Il faut également bien voir que tous les LSPs n'ont pas besoin d'être protégés par ce type de technique qui nécessite la pré-allocation des ressources. Cela peut faire l'objet d'un contrat entre l'opérateur et ses clients : protection totale du LSP, protection partagée avec d'autres LSPs, ou bien pas de protection du tout.

B.2. La protection par reroutage local (Fast-Reroute)

MPLS peut aussi assurer la protection des liens et des routeurs localement en utilisant des techniques de reroutage rapide (Fast Reroute) [RFC4090], [RFC4561]. Il devient ainsi possible d'approcher le délai de 50 ms qu'offre la reconfiguration de liens dans un réseau SDH classique.

C'est une méthode qui permet d'assurer la continuité du service en cas de panne d'un lien ou d'un nœud avec une interruption très faible du service. Le mode Fast Reroute qui est une extension de RSVP-TE assure la protection de liens (et de nœud) pour un LSP. Cela permet le reroutage local et rapide des trafics transportés par le LSP à travers un chemin contournant la panne.

La décision de reroutage est une décision locale complètement gérée par le routeur source du lien en panne. Ce dernier établit un LSP de contournement quand il reçoit une notification de panne par l'IGP ou par RSVP.

Le Fast Reroute permet de minimiser la perte de paquets due à la panne d'un lien (ou d'un nœud). Cela permet aussi de donner au routeur source du LSP le temps nécessaire à l'établissement du chemin de Backup optimal.

B.2.1. Etapes d'un Fast Reroute de lien

Nous allons décrire les différentes étapes d'un Fast Reroute de lien à travers l'exemple suivant : il s'agit d'un cœur de réseau où l'on va mettre le lien R2-R3 en panne. Le LSP entre R1 et R9 défini par les labels (37, 14, pop) utilise le lien R2-R3 et va donc être Fast Rerouté

Afin de protéger le lien R2-R3 le routeur R2 va créer un LSP de Backup (dans ce cas ce LSP passera par R6 et R7 et sera définit par la suite de labels (17, 22, pop))

La décision de reroutage des paquets revient entièrement au routeur R2. Quand celui ci reçoit la notification de la panne du lien R2-R3 il déroute les paquets sur le LSP de Fast Reroute. Cela est fait de façon très simple en insérant le label 17 aux paquets IP destiné au routeur R3 (ceci bien sur après l'opération de swap des deux labels 37 et 14).

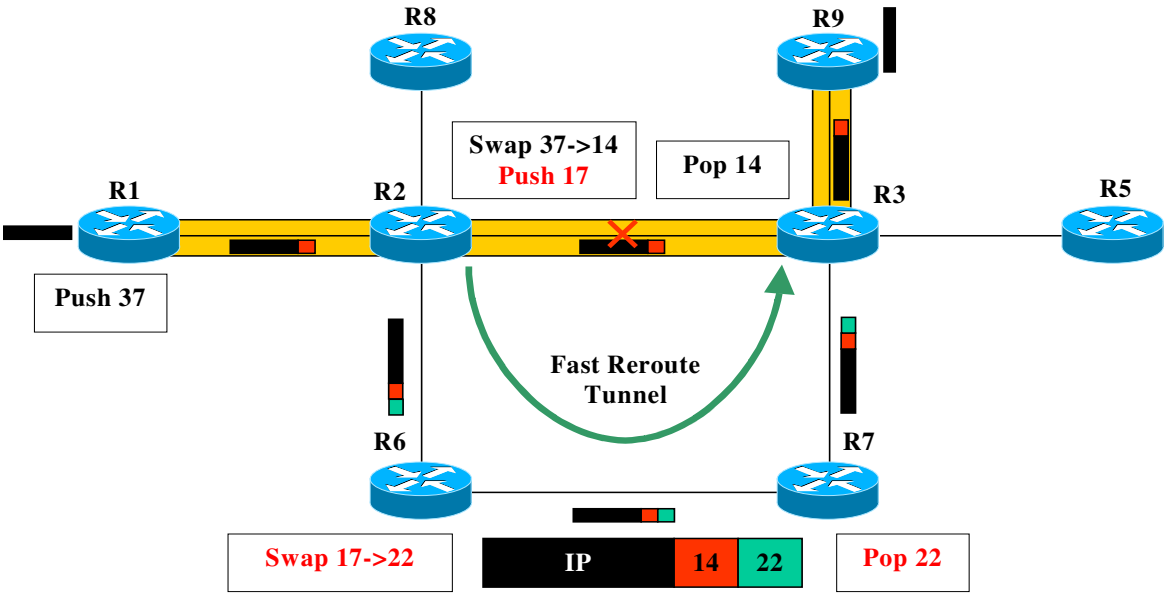


Figure 45: Fast Reroute du lien R2--R3

B.2.2. Etapes d'un Fast Reroute de nœud

Le Fast Reroute de nœuds est la généralisation du cas panne de liens. Il s'agit de créer des LSPs de détour pour tout couple de nœuds périphériques au nœud en panne et véhiculant du trafics MPLS.

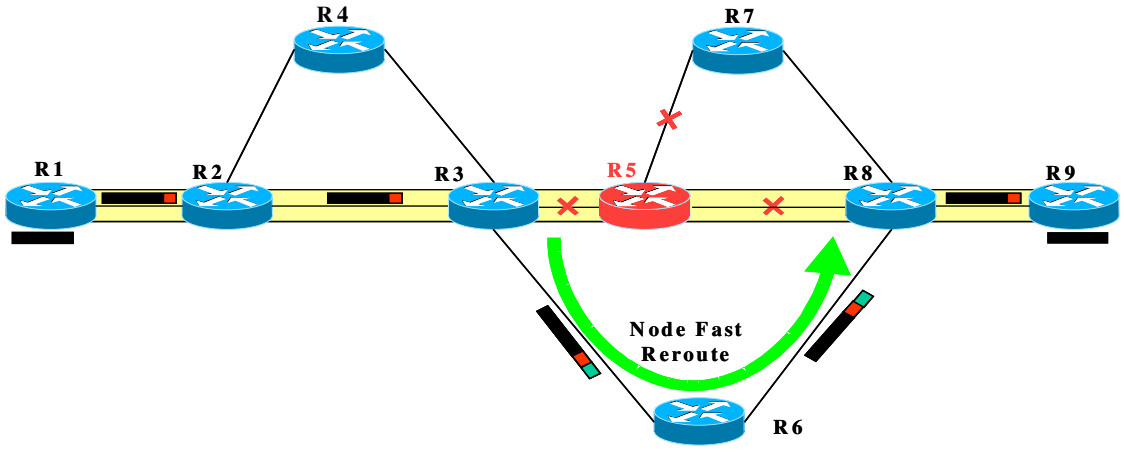


Figure 46: Fast Reroute du Nœud R5

La Figure 29 illustre le cas de la panne du routeur R5 il faudra considérer les couples suivant : (R3, R8) , (R3, R7), (R8, R3) , (R7, R3), (R7, R8) , (R8, R7)

Une fois les couples déterminés on ne garde que ceux qui véhiculent des LSPs. Dans le cas présent on ne garde que le couple (R3,R8).

Les étapes suivantes utilisent le même principe que le Fast Reroute de lien. Le routeur R3 va établir un LSP entre R3 et R8 (évitant le nœud R5). Le nœud R3 déroutera ainsi tous les paquets qui devaient transiter par le nœud R5 vers le LSP de détour en ajoutant un niveau de label.

B.2.3. Intérêts de l'approche Fast Reroute

La solution Fast Reroute apporte deux gains notables au réseau IP :

- Une fiabilité accrue pour les services IP : MPLS TE avec Fast Reroute utilisent la technologie « Fail Over Time » qui s'adapte très bien avec les techniques de restauration de lien SONET. Ce qui permet un haut degré de résilience pour les trafics IP circulant dans le cœur de réseau et des services plus robustes.

- Une scalabilité importante inhérente au design du réseau : Cela est dû au fait que le Fast Reroute utilise un mapping de tous les LSPs qui transitent par un lien Fast Rerouté vers un seul LSP de Fast Reroute. Cela permet donc de borner la croissance du nombre de LSP de Fast Reroute au nombre de liens du réseaux et non pas au nombre de LSPs.

B.3. La protection multi-niveaux (Multi-Layer)

Traditionnellement l'étude de la tolérance aux pannes se base sur l'hypothèse de la panne unique qui avec l'avènement des réseaux optiques n'est plus acceptable. En effet la pannes d'une fibre optique peut impacter plusieurs liens au niveau de la topologie IP-MPLS.

Le modèle à pannes multiples basé sur la notion de SRLG (Shared Risk Link Group) ou encore SRNG (Shared Risk Node Group) que l'on peut regrouper sous un même identifiant à savoir SRRG permet une gestion plus réaliste du problème de la résilience des réseaux actuels [RFC4205], [RFC4206].

Tenir compte des contraintes imposés par les SRRG dans le routage implique de trouver des chemins disjoints tel que les liens et nœuds des différents chemins n'appartiennent pas au même SRRG.

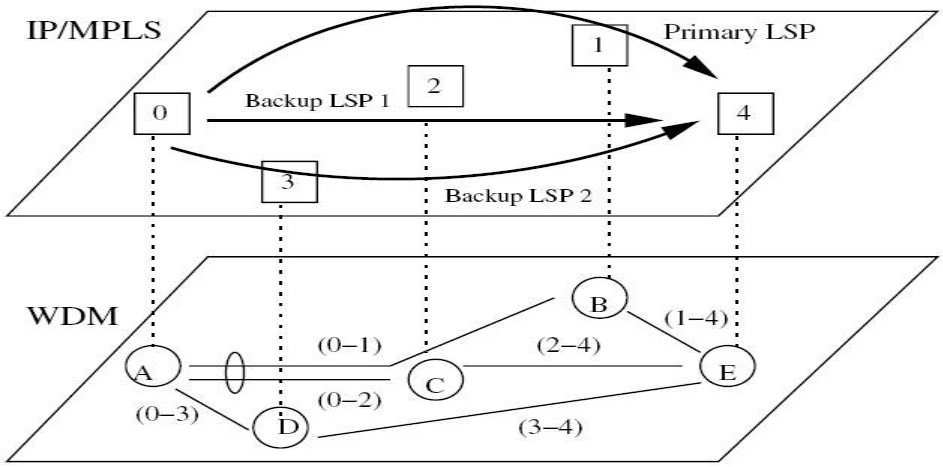


Figure 47: Sécurisation SRLG du LSP primaire

La Figure 36 montre le cas d'un réseau IP/MPLS sur WDM. Le lien (0-1) est implanté au niveau WDM par le chemin (A – C – B) . On peut voir que le LSP primaire emprunte le chemin (0 – 1 – 4) qui est nœud disjoint au niveau IP/MPLS avec le chemin de Backup (0 – 2 – 4).

Ils partagent par contre, une ressource commune au niveau WDM qui est le lien (A – C) . Le second chemin de Backup (0 – 3 – 4) est nœuds disjoints aux niveaux IP/MPLS et WDM car ils ne partagent aucune ressource commune au niveau de la topologie WDM.

C. Sécurisation des LSPs protégés

C.1. Protection par Backup

Dans cette section nous allons décrire les algorithmes que nous avons implantés pour réaliser la sécurisation des LSPs protégés.

La première approche que nous avons étudiée est la méthode élaborée par Bhandari [BAN97] qui consiste à trouver le couple optimal (chemin primaire, chemin de secours). Ce couple est optimal dans la mesure ou les deux chemins sélectionnés sont nœuds disjoints et que la somme des métriques des deux chemins est minimale.

Nous présentons par la suite une adaptation de l'algorithme ILSP-OLS-ACO pour résoudre le problème de la protection des LSPs.

Et nous terminerons par une approche mixte après avoir souligné le défaut de l'approche ILSP-OLS-ACO modifié.

C.1.1. Approche basée sur l'algorithme de bhandari

L'algorithme de Bhandari opère de la manière suivante :

Etape 1 :

Calcul du premier plus court chemin entre l'ingress et l'egress PCC1 (en rouge)

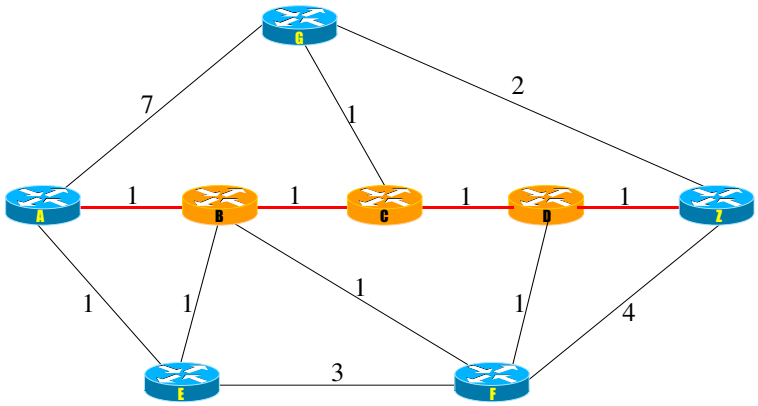


Figure 48: Calcul du PCC1

Etape 2 :

Remplacer chaque nœud appartenant au PCC1 par un couple de nœud selon la méthode VertexSplitting que nous illustrons dans la figure suivante :

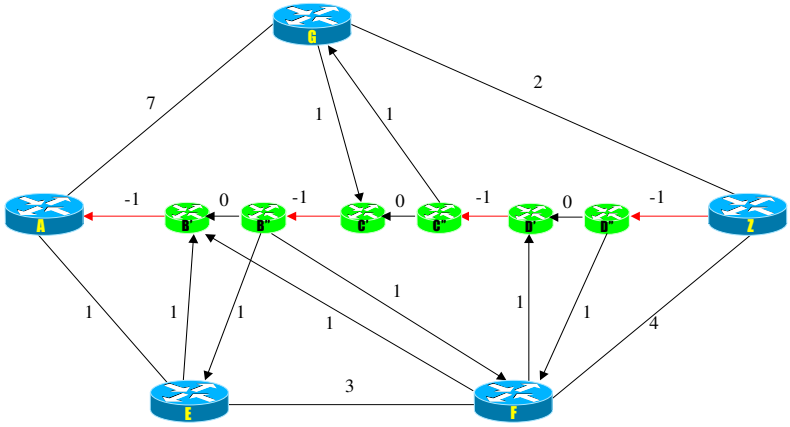


Figure 49: Application de la méthode du VertexSplitting sur le PCC1

Etape 3 :

On calcul un deuxième plus court chemin PCC2 (en bleu) à l'aide d'un algorithme de Dijkstra modifié qui permet de tenir compte de métriques négatives.

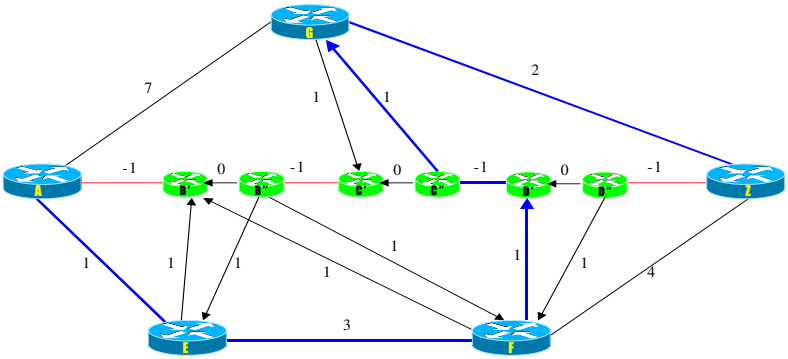


Figure 50: Calcul du PCC2

Etape 4 :

On retranscrit le chemin PCC2 dans le graphe de départ et on enlève les liens en commun.

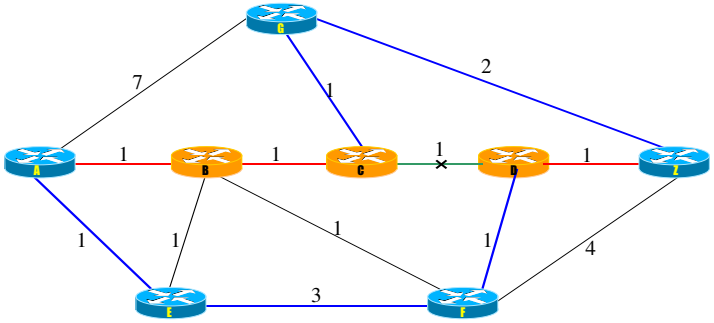


Figure 51: Retranscription du PCC2 dans le graphe de départ

Etape 5 :

Si les chemins PCC1 et PCC2 sont disjoints, l'étape 5 n'a pas lieu d'être. Sinon on combine les chemins PCC1 et PCC2 pour obtenir PCC1* et PCC2* dont la somme des métriques est minimale.

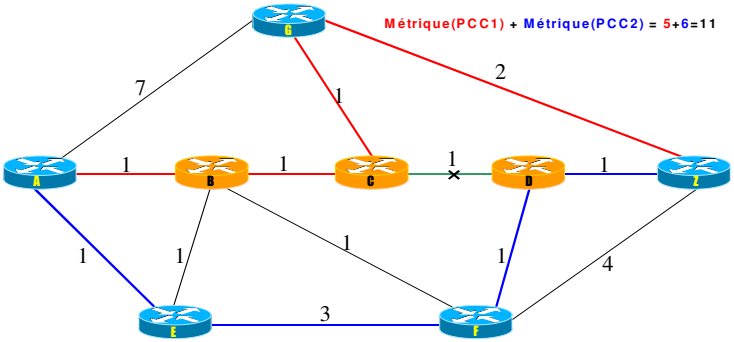


Figure 52: Construction des chemins PCC1* et PCC2*

Le principe de construction des chemins PCC1* et PCC2* est assez simple. il suffit de parcourir les liens de PCC1 et de PCC2 jusqu'à ce que l'on rencontre un lien commun. Recopier la liste des liens de PCC1 avant ce lien commun dans PCC1* et celle de PCC2 dans PCC2* . Inverser le rôle de PCC1 et de PCC2 à chaque fois que

l'on rencontre un lien commun (les liens de PCC1 seront copiés dans PCC2* et inversement).

Cette méthode permet donc d'obtenir le couple optimal chemin primaire chemin de secours nœuds disjoints. Cependant il est inconcevable de l'utiliser dans le cadre de réseau à taille réelle. En effet les différentes étapes sont gourmandes en temps de calcul spécialement les étapes 3 et 4. C'est pourquoi nous avons pensé à une approche basée sur la méthode ILSP-OLS-ACO pour résoudre ce problème.

C.1.2. Approche basée sur la méthode ILSP-OLS-ACO

Dans ce qui suit, nous allons décrire une extension de l'algorithme ILSP-OLS-ACO qui permet de calculer des chemins de secours pour les LSPs protégés. En effet pour ces LSPs on veut assurer la continuité du service en cas de panne d'un lien ou d'un nœud.

Dans ce cas un chemin de secours doit être calculé pour pouvoir y basculer les trafics en cas de pannes sur le chemin primaire. Les chemins primaires et de secours ne doivent partager aucune ressource. Ils doivent donc être nœuds-disjoints.

ILSP fonctionne toujours en R itérations, à la différence prés que l'itération j se déroule de la façon suivante :

Soit $\Omega_j = \{k \mid k = 1...K\}$ l'ensemble des LSPs à traiter au début de l'itération j,

- ✓ Choisir un LSP $k \in \Omega_j$ selon une loi uniforme. Calculer le PCC $P_{j^*}^k$ au sens de la métrique $\Gamma(x)$ pour la demande (d^k/R) d'affinité compatible et marquer le LSP ;
- ✓ Faire $\Omega_j = \Omega_j - \{k\}$;
- ✓ Si le LSP *k* n'est pas protégé alors $P_k := P_k \cup \{P_{j^*}^k\}$. Sinon vérifier l'existence d'un disjoint pour le chemin $P_{j^*}^k$, faire $P_k := P_k \cup \{P_{j^*}^k\}$ si et seulement si il existe au moins un chemin nœuds disjoints ;
- ✓ Si le chemin $P_{j^*}^k$ á été rajouté mettre à jour la charge de tous les liens lui appartenant en propageant la fraction de LSP (dk/R) ;
- ✓ Enlever la marque du LSP *k ;*

A la fin de cette version d'ILSP, tous les LSPs ont au plus *R* chemins candidats. Pour chaque chemin candidat, si le LSP est protégé, il existe au moins un chemin nœuds-disjoints.

Cependant, il se peut que pour certains LSPs, aucun chemin candidat ne soit retenu (absence de disjoint). Nous avons pris le choix de ne pas router ces LSPs (la contrainte de protection est une contrainte forte), ils ne sont pas pris en compte dans la suite de l'algorithme.

Les deux étapes suivantes à savoir ***OLS*** et ***ACO*** se déroulent comme décrit précédemment. Vient alors une quatrième partie qui consiste à calculer les chemins de secours optimaux pour les LSPs protégés.

L'optimisation simultanée du LSP primaire et du LSP de secours est un problème trop complexe à résoudre. Très peu d'études se sont attaquées à celui-ci de cette manière.

Nous avons choisi d'optimiser les LSPs de secours après avoir fixé les LSPs primaires. Cela permet de simplifier un peu le problème. L'idée étant de proposer une méthode simple et efficace capable de fournir des LSPs de secours optimaux (par rapport aux LSPs primaires retenus).

Cette quatrième partie fonctionne de la manière suivante :

- ✓ Trier les liens *u* dans l'ordre décroissant de bande passante protégée* (ou un autre critère).
- ✓ Pour chaque lien *u* (considérés dans l'ordre défini précédemment) faire
  - Enlever le lien *u* du réseau (simulation de panne),
  - Enlever tous les LSPs qui passent par le lien *u,*
  - Notons Ω_u l'ensemble des LSPs protégés que l'on vient de retirer du réseau, trier Ω_u dans l'ordre décroissant de priorité, puis par ordre décroissant de bande passante,
  - Pour chaque LSP protégé $k \in \Omega_u$ (considérés dans l'ordre défini précédemment) faire :
    - Calculer le PCC nœuds-disjoints $\overline{P}_*^k$ par rapport à la métrique Γ choisie, en utilisant l'algorithme Dijkstra. Le chemin $\overline{P}_*^k$ sera le LSP de backup du LSP primaire P_*^k ,
    - Mettre à jour les charges des liens en propageant la demande d^k,
    - Faire $\Omega_u = \Omega_u - \{k\}$,
  - Depropager tous les LSPs concernés par la panne du lien *u* de leur chemin de backup $\overline{P}_*^k$,
  - Repropager tous les LSPs concernés par la panne du lien *u* sur leur chemin primaire P_*^k .

(*) La bande passante protégée du lien *u* est la somme des bandes passantes requises par tous les LSPs protégés qui passent par ce lien ($u \in P_*^k$).

A la fin de cet algorithme chaque LSP protégé possède un chemin de secours nœuds-disjoints. Cet algorithme est très rapide et permet de calculer, en plus des chemins de secours, le coût de la panne de chaque lien.

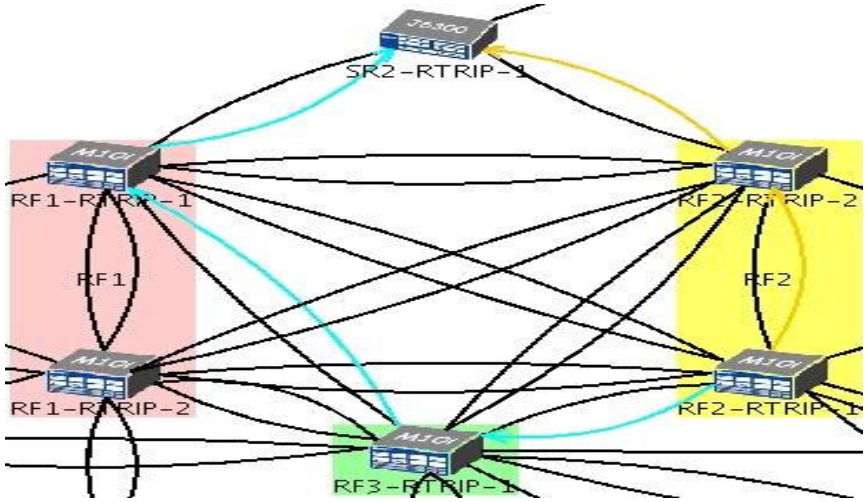


Figure 53: Sécurisation du LSP RF2-RTRIP1 vers SR2-RTRIP1

Les LSPs de secours, obéissent aux mêmes contraintes que les chemins primaires, à la différence près que la QoS du chemin de secours peut être parfois légèrement dégradée par rapport à celle du chemin primaire. Cela reste toutefois tolérable car les trafics du LSP ne transitent par le chemin de secours que pour une courte durée.

Cette durée équivaut au temps de restauration de la panne. Cet algorithme prend en compte le cas de plusieurs pannes simultanées (de liens) dans le cas ou des SRRG sont définis.

L'allocation de bande passante pour les LSPs de secours utilise le principe «shared memory». Cela veut dire que cette bande passante, si elle est allouée, permet de sécuriser la panne de plusieurs LSPs en même temps (elle peut supporter plusieurs types de pannes n'ayant pas lieu simultanément).

Cela se traduit par le fait que sur une interface ou un lien la bande passante allouée pour la sécurisation est le maximum des bandes passantes des LSPs à protéger pour des pannes différentes et non pas la somme de celles-ci.

Chaque LSP est protégé donc selon le mode (1:1) et son chemin de secours, quand il est alloué, est utilisé par d'autre trafics (moins prioritaires) tant qu'il n'y a pas de panne. Cette méthode permet une meilleur gestion des ressources réseau.

C.1.3. Approche mixte

L'étude de l'algorithme modifié de ILSP-OLS-ACO nous à permis de nous rendre compte que c'est un algorithme efficace pour le placement des LSPs primaires et qu'il fournit de bonnes solutions pour les routes de protection.

Cependant nous avons été confronté dans certains cas à une faille de notre algorithme que nous illustrons dans la figure suivante.

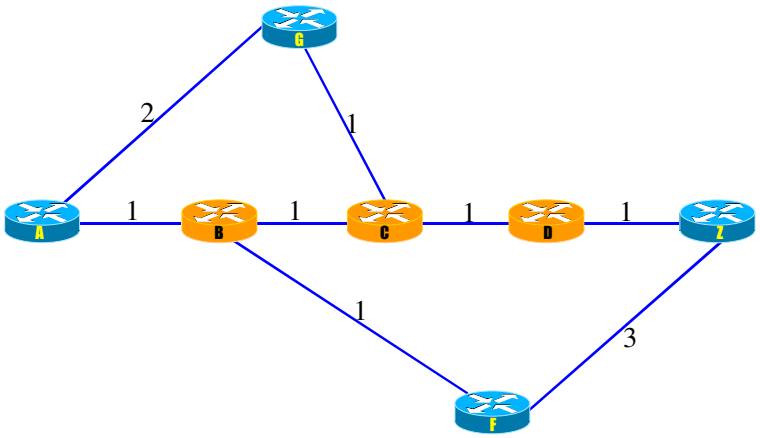


Figure 54: Illustration d'un cas critique pour ILSP-OLS-ACO

Si le chemin (A – B – C – D – Z) est choisi comme PCC il n'est plus possible de trouver un chemin nœuds disjoints. Alors qu'un solution est possible, en effet les chemins (A – G – C – D – Z) et (A – B – F – Z) sont nœuds disjoints.

Nous rappelons que les métriques que nous utilisons pour ILSP-OLS-ACO sont dynamiques et dépendent entre autre de la bande passante résiduelle du lien.

Donc le cas qui met en défaut notre approche intervient quand le PCC primaire d'un LSP ne possède pas de disjoint et qu'au cours des itérations d'ILSP aucun autre LSP ne modifie la métrique de ce PCC (ce qui permettrait d'en générer un autre puisque la métrique ne serait plus minimale).

C'est un cas certes peu courant mais que nous proposons de résoudre par une méthode mixte qui combine ILSP-OLS-ACO et Bhandari.

Le principe est simple : nous utilisons la version modifiée de ILSP pour générer les couples de chemin primaire / chemin de secours. Cependant si au cours d'une itération d'ILSP, aucun couple de chemin primaire / de secours n'est généré pour un LSP, nous faisons appel à la méthode Bhandari pour générer ce couple. Les itérations suivantes, si on se retrouve dans la même cas de figure, nous réutilisons le chemin Bhandari généré précédemment et nous propageons le quantum de trafic sur ce chemin comme s'il avait été généré par ILSP.

Cela permet d'obtenir le meilleur des deux algorithmes : rapidité et efficacité d'ILSP-OLS-ACO modifié et garantie d'avoir un couple de (chemin primaire , chemin de secours) optimal par Bhandari.

C.2. Protection par Reroutage local (Fast Reroute)

La protection par Fast Reroute permet de gérer une situation de panne avec des temps de réponse rapides dans la mesure où elle propose une solution de contournement de la panne au niveau du routeur amont de celle-ci. Il s'agit donc de simuler ce fonctionnement en traitant le cas de tous les éléments du réseaux susceptibles de provoquer une panne.

L'algorithme de protection par Fast Reroute va traiter tous les éléments signalés comme défaillants et puis va calculer un chemin de contournement pour les pannes de liens et pour chaque couple (nœud en amont – nœud en aval) pour le cas d'une panne de nœud.

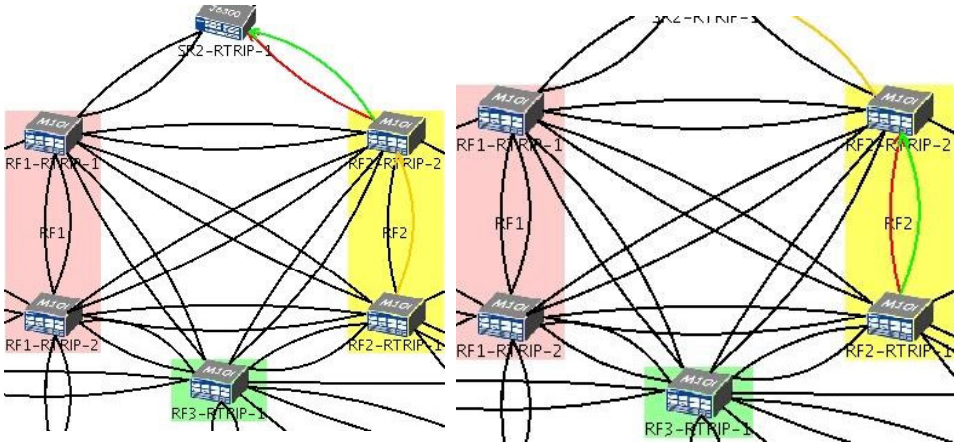


Figure 55: Exemple de Fast Reroute de liens

L'exemple de la Figure 55 permet d'illustrer les chemins de contournements (en rouge) calculés par l'algorithme Fast Reroute dans le cas d'une panne de lien (en vert).

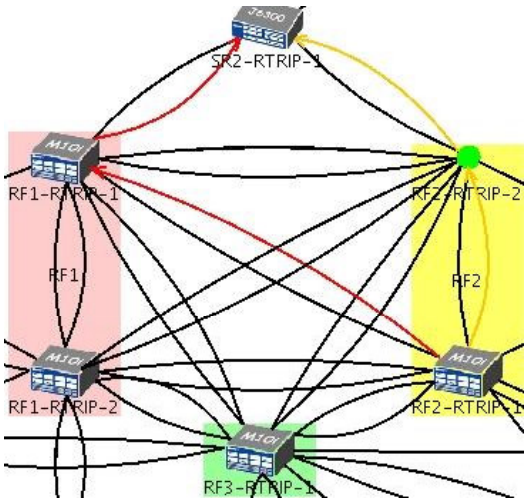


Figure 56: Exemple de Fast Reroute de nœud

Celui ci illustre le chemin de contournement de la panne (en rouge) du nœud RF2-RTRIP2 .

Nous allons illustrer dans ce qui suit le principe de fonctionnement de l'algorithme qui calcule le Fast Reroute d'un lien. Nous rappelons que le Fast Reroute d'un nœud n'est qu'une généralisation du précédent cas dans la mesure ou cela revient à considérer un Fast Reroute de lien entre le nœud en amont du nœud en panne et le nœud en aval de celui ci.

L'algorithme se déroule de la manière suivante :

- Identifier les LSPs en transit sur chaque lien,
- Trier les liens suivant un critère donné (bande passante à sécuriser, nombre de LSPs en transit, …),
- Parcourir les liens du réseau,
  - Si le lien doit être Fast Rerouté,
    - ✓ Mettre en panne le lien,
    - ✓ Re-router les trafics OSPF/ISIS qui n'empruntent pas de LSPs,
    - ✓ Créer deux LSPs (un dans le sens Forward l'autre pour le sens Backward) dont la bande passante est la somme des bandes passantes des LSPs Fast Reroutable dans chaque sens,
    - ✓ Dépropager les LSPs non Fast Reroutable de la topologie MPLS et les repropager par OSPF/ISIS,
    - ✓ Mettre à jour les charge des liens et des interfaces au niveau MPLS,
    - ✓ Calculer le PCC du LSPs dont l'ingress est la source du lien et dont l'egress est la destination du lien dans la nouvelle topologie MPLS,

C.3. Protection multi-niveaux (Multi-layer)

La protection multi-layer est basée sur la notion de SRRG comme défini précédemment. Il s'agit de définir des groupes d'éléments qui partagent la même ressource. Ainsi la panne de cette ressource provoque par voie de fait la panne de tous le groupe.

Dans l'exemple qui suit nous avons une topologie IP-MPLS sur de l'ATM et nous définissons les SRLG par rapport aux switchs et liens ATM

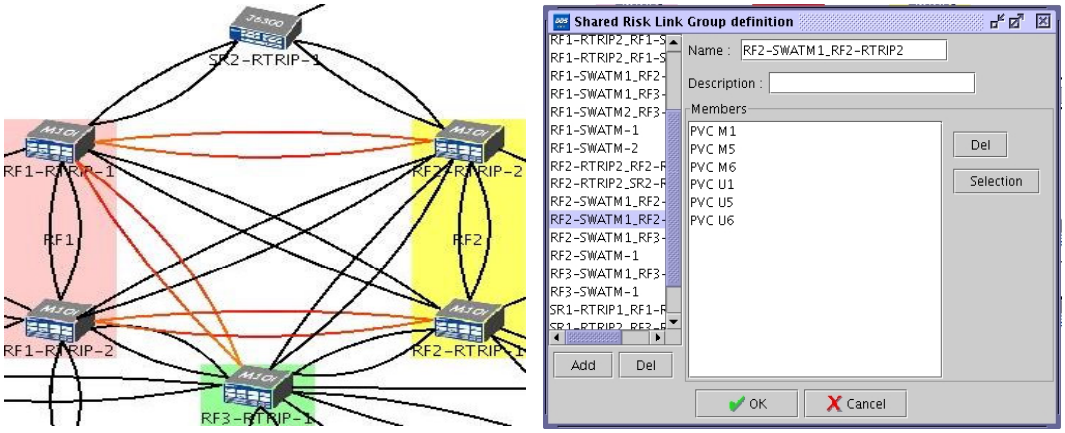


Figure 57: Définition d'un SRLG pour le lien RF2-SWATM1<-->RF2-RTRIP2

La panne du lien RF2-SWATM1 vers RF2-RTRIP2 entraîne celle aussi des PVCs {M1, M5, M6, U1, U5, U6}. C'est les liens en rouge que l'on aperçoit sur la figure précédente.

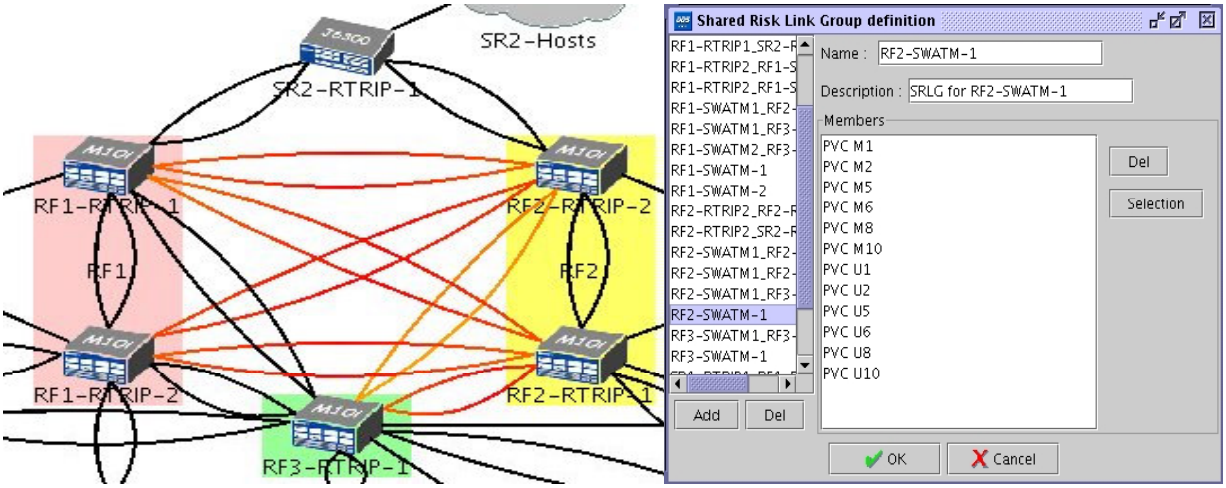


Figure 58: Définition d'un SRLG pour le nœud RF2-SWATM-1

La figure 58 illustre la définition d'un SRLG pour le switch ATM RF2-SWATM-1 qui une fois en panne affecte les liens PVC : {M1, M2,M5, M6, M8, M10, U1, U2,U5, U6, U8, U10}. C'est encore une fois les liens colorés de la figure précédente.

Donc la sécurisation d'un LSP dont le primaire emprunte l'un des PVC cités au dessus ne doit emprunter aucun des Liens PVC du même groupe pour le chemin de secours.

L'algorithme utilisé se base essentiellement sur la construction du graphe résiduel obtenu en élaguant tous les liens impactés par cette panne au travers des différents SRLG en cause.

Plus précisément lors de l'étude de la panne d'un lien L1 nous cherchons à quel SRLG il appartient. Notons S1 le premier SRLG contenant L1.

1- Mettre en panne tous les liens appartenant à S1,
2- Pour chacun des liens de S1,
    a. Parcourir tous les SRLGs du réseau,
        i. Si le SRLG courant contient ce lien mettre tous les liens du SRLG en panne.

Un fois ce graphe construit nous utilisons l'algorithme de calcul de chemin de Backup présenté précédemment.

Comme on peut le voir dans l'exemple suivant le LSP49 entre RF2-RTRIP2 et RF3-RTRIP1 doit être protégé. Nous illustrons le résultat obtenu avec et sans SRLG définis.

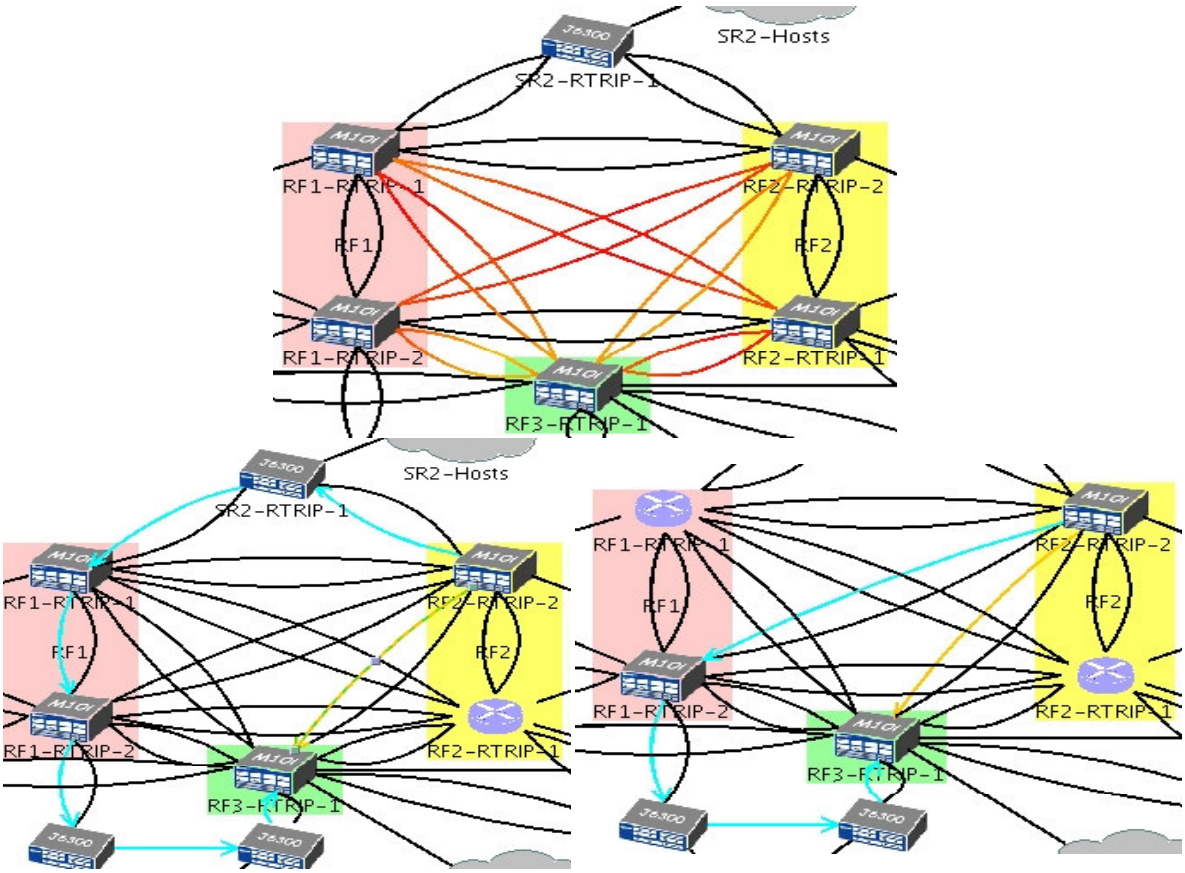


Figure 59: Calcul du Chemin de secours avec et sans SRLG

Pour le calcul du chemin de secours avec SRRG il fait tenir compte de celui qui est présent dans la figure 58 et celui dans le figure 59. On peut constater que le LSP de secours dans la partie gauche de l'image contourne tous les liens appartenant au SRLGs impactés par la panne du PVC U6.

D. Conclusion

Dans ce chapitre nous présentons les différentes techniques de sécurisation utilisées dans le cadre des réseaux MPLS. Ainsi nous présentons la technique de sécurisation de bout-en-bout (Backup), le reroutage local (Fast Reroute) ou encore une généralisation multi-couches de ces deux dernières techniques.

Nous présentons une approche basé sur la méthode ILSP-OLS-ACO pour gérer la sécurisation de bout en bout. Nous avons aussi présenté une amélioration de cette méthode qui combine ILSP-OLS-ACO et la méthode Bhandari.

Nous traitons par la suite la technique des Fast Reroute qui propose une alternative intéressante à la sécurisation de bout en bout.

Finalement nous présentons une approche multi-niveaux de la sécurisation qui permet de généraliser les deux méthodes précédentes au cas des réseaux multi-couches par le biais des SRRG.

« Les ordinateurs du futur ne devraient pas peser plus de 1,5 tonnes »

Popular Mechanics, commentaires sur l'avancée des sciences, 1949.

Chapitre 6 - CONCEPTION DE TOPOLOGIE

A. Introduction

Ces dernières années ont été marquées par l'essor des Nouvelles Technologies de l'Information et de la Communication (NTIC) pénétrant tous les secteurs d'activité: industriel, commercial, bancaire, éducation. Phénomène de mode, ou réel besoin pour les entreprises, la conséquence immédiate a été de remplacer les structures de communication existantes ou simplement de les mettre en place afin de pouvoir répondre aux nouvelles exigences des utilisateurs (communications intranet, extranet, internet,…).

Les réseaux de télécommunications sont devenus des ressources stratégiques et leur importance économique ne cesse d'augmenter. Ainsi pour faire face à la croissance et à l'augmentation de la charge des réseaux, la première idée est d'utiliser des routeurs puissants avec beaucoup de mémoire, des processeurs rapides et des lignes à hauts débits. Il va de soit que le coût de telles infrastructures peut être exorbitant.

Partant de cet état de fait, l'élaboration d'une topologie optimale nous est apparue, dans ce contexte, un point important à étudier. Les principales difficultés rencontrées vont être de minimiser le coût total du système de communication tout en garantissant une QoS globale du réseau (résilience).

Dans notre étude, l'accent sera porté sur la planification de réseaux modernes de communications. Ce domaine s'est développé considérablement au cours de ces dernières années, en raison principalement de l'accroissement considérable de la taille des réseaux.

La planification et la conception des réseaux de télécommunications représente une tâche très complexe et en règle générale, fort coûteuse. L'équipe de planification doit passer en revue les besoins déjà existants ou anticiper les besoins futurs, les coûts des différentes composantes des systèmes, les contraintes imposées aux performances, la fiabilité, la capacité d'adaptation aux évolutions, le service de contrôle de la qualité, etc.

Ainsi la conception d'un WAN (Wide Area Network) est un processus dans lequel des dizaines de sites aux caractéristiques différentes sont connectés afin de satisfaire à certains standards de fiabilité et de performance, et ce, à des coûts minimes. Ce processus de conception comprend la localisation des sites terminaux, localisation des concentrateurs, la conception du réseau de cœur (Backbone), les procédures de routage, ainsi que le dimensionnement des lignes et nœuds.

L'une des questions clés de la conception d'un WAN est la grande complexité du problème. Ainsi même en décomposant le problème global, les sous problèmes auxquels on aboutit ne sont pas triviaux. Etant donné l'importance des investissements, si l'on réussit à réduire légèrement les coûts de quelques points de pourcentage, tout en assurant la même qualité des services, il sera possible de dégager des bénéfices économiques considérables.

Une topologie globale de réseau WAN peut normalement être décomposée en deux éléments principaux: le réseau d'accès et le réseau dorsal. Ces composantes présentent des propriétés très différentes et entraînent donc des problèmes de conception spécifiques bien qu'étroitement interdépendants. Cette spécificité constitue à elle seule une grande partie de la complexité du problème global.

Un réseau d'accès WAN comprend un certain nombre de sous-réseaux d'accès, dotés de topologies de type arborescent; et les sommets de concentration des flux permettent d'en diminuer les coûts. Ces flux intégrés atteignent le cœur de réseau, qui a une topologie maillée, afin de remplir les critères de sécurité, de fiabilité et de performance. En conséquence, le cœur de réseau est d'habitude composé de lignes de communication de haute capacité, tels les liens en fibre optique.

En règle générale, cette caractéristique topologique du WAN est valable dans le cas d'un réseau basée sur un système datagramme (c'est le cas de la technologie IP) ainsi que dans le cas de la commutation de circuits (cas du réseau téléphonique actuel et d'un certain nombre d'autres technologies, notamment X25, Frame Relay ou ATM).

En général, l'équipe de conception gère un nombre considérable de données qui lui permettent de proposer un modèle satisfaisant les conditions requises. Elle dispose par exemple d'informations sur l'ensemble des positions des sites terminaux (les entreprises clients, les abonnés au service, etc.) et sur les caractéristiques (le plus souvent, estimées) des flux inter-sites (volume, comportement dans le temps, etc.). Il existe également des informations relatives à la fiabilité, la connectivité, la sécurité et la disponibilité du réseau.

Quant à l'aspect des composantes du réseau, le concepteur dispose d'une liste d'équipements possibles selon la nature, les caractéristiques et les coûts du réseau en question. La nature technique du réseau examiné conduit à des procédures spécifiques de routage qui devraient être prises en considération pour assurer des solutions efficaces, voire, dans la mesure du possible, optimales.

En général, il y a bien d'autres données complémentaires, notamment quels sont les sites les plus aptes, ou les moins adaptés pour installer des concentrateurs, quels sont les sites du cœur de réseau qui devraient être munis de commutateurs (switch), quelles sont les caractéristiques particulières ou les restrictions de sécurité applicables à certains flux, etc.

Sur la base de cet ensemble de données, le concepteur doit spécifier les topologies du réseau d'accès et du cœur de réseau, ainsi que les caractéristiques des différents sites et connexions, le routage des trafics, etc. Ce processus aboutit à l'optimisation de modèles spécifiques de solution du problème global de conception tant des sous-réseaux que du WAN. Cet ensemble de problèmes comprend en règle générale l'évaluation des performances, de la fiabilité, etc.

Notre objectif est donc de proposer des algorithmes performants permettant de générer des architectures de réseau optimales répondant aux critères de résilience défini par l'ingénieur réseau.

B. Etat de l'art

L'étude des problèmes de conception de topologies optimales pour les réseaux d'accès remonte aux années 60 avec l'introduction des problèmes 1-median et p-median par Hakimi [HAK64], [HAK65]. Plusieurs modèles ont vu le jour depuis. Ces modèles se différencient par la façon dont ils évaluent le réseau obtenu. Certains modèles mettent l'accent sur le coût d'installation du réseau (minimisation du coût des installations), le coût de fonctionnement (minimisation du coût d'installation et du délai moyen dans le réseau), la fiabilité, la capacité à gérer les surcharges.

La modélisation de la conception d'un WAN au moyen de la formulation d'un seul problème d'optimisation mathématique est extrêmement complexe du fait de l'interdépendance de ses nombreux paramètres. Dès lors, la conception d'un WAN est souvent subdivisée en plusieurs sous-problèmes différents. Un bon exemple d'une démarche possible de décomposition du processus de conception WAN est présenté dans [PRI99].

La méthodologie présentée dans [PRI99] peut être une référence utile sur la manière de décomposer le processus de conception de la topologie du WAN en plusieurs sous-problèmes. Concernant d'autres travaux connexes, portant sur la conception optimale d'un réseau hiérarchique à plusieurs niveaux, le lecteur peut consulter les références [BAL94,MAG94,MIR98,SUN92,CHA01,SAN00,CRU99,JAE92,KOS02, THO02]. Elles sont pour l'essentiel centrées sur la conception de la topologie et passent également en revue plusieurs aspects relatifs au dimensionnement du réseau.

D'autres problèmes associés à ce volet peuvent être consultés dans les références [ALT92,BOO77,GAV86,GAV92a,GAV92a,PIE97,TRA98], où les auteurs proposent un certain nombre de modèles de conception de topologies de réseaux à faible coût avec des contraintes supplémentaires, notamment la tolérance aux failles et des restrictions de performance, et étudient en outre, pour certain d'entre eux, le dimensionnement des composantes du réseau.

Les auteurs [ALT93,AND98,BER04,FRA99,GAV86,GAV91,GOU97,GIR01,LUK99, MAT00,RAN01,MAH01] proposent différents algorithmes approximatifs de la conception topologique des réseaux d'accès locaux et de grande échelle. Les auteurs de ces ouvrages se fondent sur des approches différentes et prennent en considération des paramètres et des restrictions différents, y compris les aspects ci-après: la conception du réseau d'accès est réservée à des topologies spécifiques, le nombre de concentrateurs à installer est limité, les composantes du réseau sont dimensionnées, etc.

Les techniques de résolution utilisées dans ces travaux comprennent: la relaxation Lagrangienne combinée avec les méthodes de sous-gradient, relaxation de programme linéaire, heuristique Lagrangienne, heuristique gloutonne, Branch-and-Bound combiné avec relaxation Lagrangienne, Branch-and-Bound avec décomposition de Benders, Réseaux de Neurones, recherche tabou, algorithmes génétiques, méthode Branch-and-Bound combinée avec l'algorithme de Ford-

Fulkerson, procédure Dual-Based , procédure de Dual-Based « lower bounding » incorporée à un algorithme Branch-and-Bound, etc…

Nous allons adopter dans ce qui suit la vision décomposée du problème de conception de topologie des WAN. Nous nous intéressons plus particulièrement à la conception de la topologie du réseau d'accès. Toutes les études citées auparavant concentrent les coûts au niveau des liens et des sites. Cela est du à l'importance des coûts d'ouverture de sites ainsi qu'a ceux relatifs à la mise en place des tranchées.

Cependant cette vision n'est plus d'actualité car on creuse de moins en moins de tranchées vu les millions de kilomètres de fibre optique déjà mise en place (160 millions au USA (2005), 25 millions au Japon (2005), 2 millions à Paris (2007)). L'originalité de l'étude que nous proposons met l'accent sur le coût des équipements qui devient prédominant dans le coût final. La modélisation que l'on propose permet de résoudre le problème de l'affectation du client « c » à la carte « j » du routeur « r » au niveau du site « s ». Ce problème est une forme beaucoup plus complexe que le problème traditionnellement étudié mais plus proche de la réalité des opérateurs.

Vu la nature NP-difficile du problème, il est encore possible d'améliorer les solutions pratiques appliquées par l'industrie actuellement. Notre motivation vient du besoin de créer des algorithmes approximatifs efficaces pour résoudre ces problèmes de conception topologique hautement combinatoires. Nous croyons en ce sens que la conception d'heuristiques puissantes est d'une grande importance stratégique, et qu'elles peuvent facilement être intégrées dans des outils. Précisons que même une toute petite réduction de ces coûts peut représenter plusieurs millions d'euros d'économies, pour les opérateurs.

Dans ce qui suit nous introduisons des modèles d'optimisation combinatoire pour définir formellement la conception topologique des réseaux d'accès, et nous proposons plusieurs algorithmes pour la résolution de ce problème.

C. Formulation du problème de conception du réseau d'accès

La conception d'un réseau est un processus structuré qui à partir des volumes de trafic transmis sur le réseau par les utilisateurs et de l'emplacement des sites potentiels de raccordement permet :

- de choisir les technologies qui répondent aux besoins qui ont été identifiés ;
- de créer la conception finale de la topologie réseau qui répond aux critères de performances, selon les besoins des utilisateurs et les coûts.

Pour ce faire il faut se poser plusieurs questions fondamentales

- Combien de sites sont nécessaires pour connecter les clients ?
- A quel endroit faut il les placer ?
- Quel type de routeurs et combien en faut il dans chaque site ?
- Comment affecter les clients aux sites retenus ?

- Comment connecter les clients aux sites (étoiles, arbre) ?
- Quel est la capacité des liens que connectent les clients au sites ?
- Comment les sites sont connectes entre eux ?
- Quel est la capacité des liens reliant les sites ?
- Quel est le routage dans le cœur de réseau ?

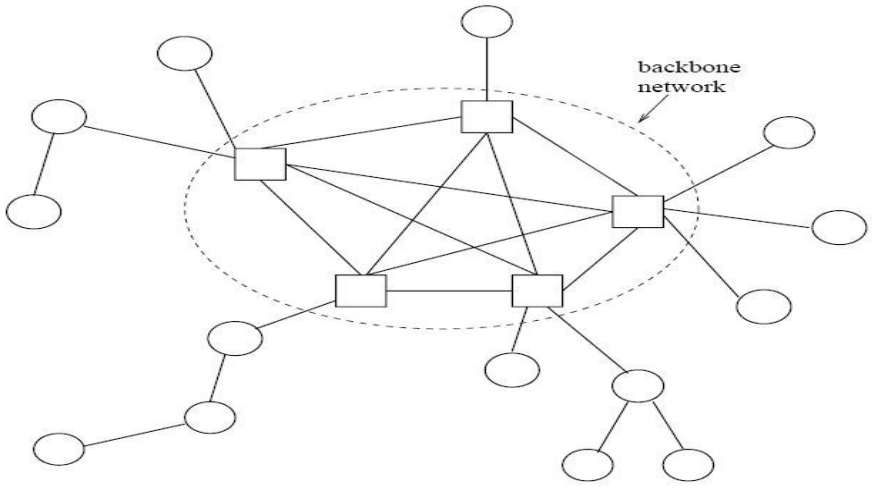


Figure 60: Réseau full-mesh en cœur / en arbre pour l'accès

Il est difficile de répondre au travers d'un modèle unique à toutes les questions précédentes. En général le processus de conception de topologie est découpé de la sorte .

- On fixe le nombre de sites nécessaires
- On traite la conception de la topologie d'accès
- On traite la conception de la topologie du cœur de réseau

De ce fait les données de départ d'un problème de conception de réseau d'accès sont la matrice de trafic des clients à connecter au réseau, les points de présence des clients et la liste des sites potentiels (points de raccordements) .

Le concepteur de la topologie exige la sécurisation de certains clients. Pour ce type de clients une affectation représente le fait qu'il soit connecté à un site primaire et à un site secondaire.

C.1. Formulation Mathématique du problème

On considère N nœuds clients à raccorder par des liens de communication à des routeurs situés dans K sites potentiels. Le problème consiste à déterminer les sites à utiliser, leurs équipements et les clients qui leur sont connectés pour construire un réseau à moindre coût.

Pour chaque nœud client $i = 1,...,N$, on connaît sa demande d_i. Cette demande correspond à la capacité (symétrique ou asymétrique) du lien permettant de

raccorder le client au réseau. Typiquement les valeurs de ces demandes seront : 512Kbps, 2Mbps, 10 Mbps, 100 Mbps, 1 Gbps, etc.

L'affectation des clients aux sites est un sous problème qui doit être résolu en même temps, néanmoins lorsque celle ci est connue on sait déterminer l'équipement nécessaire dans chaque site.

Comme on connaît également le coût de chacun des sites, on connaît le coût global de la solution obtenue. En d'autres termes, l'affectation des clients aux sites permet de déterminer quels sont les sites utilisés et quel est leur équipement.

Une solution du problème est donc un vecteur $x = \left[x_{ij} \right]$ tel que,

$$x_{ij} = \begin{cases} 1 & si\ le\ client\ i\ est\ connecté\ au\ site\ j \\ 0 & sinon \end{cases} \tag{6.1}$$

Une solution x est admissible si tout client i est affecté à exactement un site j:

$$\sum_{j=1}^{K} x_{ij} = 1 \qquad i = 1,...,N \tag{6.2}$$

C.1.1. Coût d'une solution

Le coût d'une solution x représente la somme des coûts des sites utilisés, des coûts d'équipements à installer dans chaque site pour supporter la demande, et des coûts des liens permettant de raccorder les clients aux sites de concentration.

C.1.1.1. Coût des sites

Soit L_j le coût du site j. C'est le coût d'achat ou de location des locaux dans lesquels se trouveront les équipements réseaux permettant de concentrer le trafic venant des nœuds clients.

On définit la variable de décision u_j , $j = 1,...,K$ de la façon suivante :

$$u_j = \begin{cases} 1 & \sum_{i=1}^{N} x_{ij} \geq 1 \\ 0 & sinon \end{cases} \tag{6.3}$$

La variable u_j indique si le site j est utilisé ou non dans la solution x.

Le coût total des sites associé à la solution x s'écrit alors,

$$\Gamma_S(x) = \sum_{j=1}^{K} L_j \cdot u_j \tag{6.4}$$

C.1.1.2. Coût des liens

Pour la solution x, si $x_{ij} = 1$, cela signifie qu'il faut raccorder le nœud client i au site j par un lien de communication. Le coût de ce lien sera d'autant plus fort que la distance (euclidienne) entre i et j sera grande et que la demande d_i du client i sera grande.

Notons l_{ij} la distance à vol d'oiseau en km entre le nœud client i et le site j.
Typiquement, une offre opérateur va définir des seuils de distance $l_1 \prec l_2 \prec \cdots \prec l_M$ tels que le coût du lien (i, j) sera donné par,

$$c_{ij} = \alpha_k(d_i) + \beta_k(d_i) \cdot l_{ij} \qquad si \quad l_{k-1} \prec l_{ij} \prec l_k \tag{6.5}$$

$\alpha_k(d_i)$: coût d'ouverture du service,
$\beta_k(d_i)$: coût kilométrique.

Avec par convention $l_0 = 0$ et $l_{M+1} = \infty$. Les distances entre les nœuds clients et les sites potentiels étant connues à l'avance, les coûts c_{ij} , pour $i = 1,...,N$ et $j = 1,...,K$ sont des constantes du problème

Le coût global des liens dans la solution x s'écrit,

$$\Gamma_L(x) = \sum_{i=1}^{N} \sum_{j=1}^{K} c_{ij} \cdot x_{ij} \tag{6.6}$$

C.1.1.3. Coût des équipements

Si le site j est sélectionné dans la solution x, il faut y installer des équipements permettant de supporter le trafic en émission et réception des clients raccordés à ce site.

Le coût des équipements installés au site j dépend du nombre de clients raccordés à ce site et des demandes de ces clients .

Notons $E_j(x)$ le coût des équipements à installer sur le site $j = 1,...,K$ pour l'affectation x.

L'analyse de ce coût sera faite dans la section C.1.2. De manière générale, on fera l'hypothèse d'une économie d'échelle en faveur des équipements de grande capacité. Autrement dit, on a intérêt à concentrer le trafic en termes de coût des équipements.

Le coût global des équipements s'écrit,

$$\Gamma_E(x) = \sum_{j=1}^{K} E_j(x)$$

C.1.1.4. Coût global d'une solution

Le coût global de la solution x s'écrit:

$$\Gamma(x) = \sum_{j=1}^{K} L_j \cdot u_j + \sum_{i=1}^{N}\sum_{j=1}^{K} c_{ij} \cdot x_{ij} + \sum_{j=1}^{K} E_j(x) \qquad (6.7)$$

On remarque que $\Gamma_S(x)$ et $\Gamma_E(x)$ poussent à la concentration, c'est à dire à la minimisation du nombre de sites en retenant ceux qui coûtent le moins cher. En fait, si les coûts de distance sont négligeables, les coûts de sites et d'équipements vont conduire vers une solution centralisée sur le site de moindre coût.

Au contraire, les coûts de distance conduisent vers une solution distribuée dans laquelle chaque client est affecté au site le plus proche. Si les coûts de site et d'équipements sont négligeables, on aura donc une solution au plus près.

En général, la solution obtenue devra être un compromis entre une solution centralisée et une solution au plus près en fonction des poids relatifs des différents coûts.

C.1.2. Analyse des équipements nécessaires dans un site

Considérons un site $j = 1,...,K$ et une affectation des clients aux sites $x = [x_{ij}]$. Si dans cette solution des clients sont affectés au site j, il faut dimensionner les équipements de ce site pour supporter la demande en trafic de ces clients. Ces équipements correspondent à des routeurs et à des cartes que l'on place sur ces routeurs.

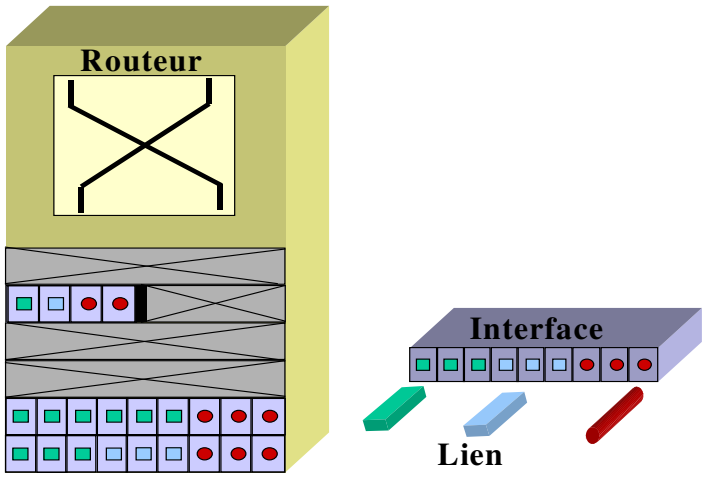


Figure 61: Modélisation Routeur, Interface, Lien

On suppose qu'on dispose de R + 1 modèles de routeur et I + 1 modèles d'interface.

Le modèle de routeur $r = 0,...,R$ est caractérisé par,

- s_r : le nombre de slots disponibles pour accueillir des cartes,
- T_r : représente le débit maximal que peut supporter le bus interne du routeur,
- ψ_r : représente le coût du châssis.

Le modèle de routeur r = 0 correspond à un artifice de modélisation. Il est caractérisé par $(s_0 = 0),(T_0 = 0),(\psi_0 = 0)$. Autrement dit, ce modèle n'a aucune capacité de transmission et ne coûte rien. Ce modèle sera appelé le modèle de routeur nul. Les autres modèles seront dits effectifs.

Un modèle de carte $t = 0,...,l$ est caractérisée par,

- p_t : le nombre de ports de communication disponibles,
- r_t : représente le débit crête de chaque port,
- φ_t : représente le coût global de la carte.

Les débits r_t par port de communication correspondent aux demandes d_i des clients. ainsi, nous supposons que pour chaque client *i*, il existe un et un seul modèle de carte *t* tel que $(d_i = r_t)$. On notera Φ_t l'ensemble des clients *i* vérifiant cette relation.

De la même façon que dans le cas des routeurs, le modèle de carte t = 0 est caractérisé par $(p_0 = 0),(r_0 = 0),(\varphi_0 = 0)$. Ce modèle, appelé dans la suite le modèle de carte nul, a donc un coût nul et une capacité de transmission également nulle. Les cartes dont le modèle sera non nul seront dites effectives.

On supposera que chaque site $j = 1,...,K$ dispose de M routeurs, y compris les routeurs nuls. Les contraintes du problème imposeront par exemple qu'un site non utilisé $(u_j = 0)$ n'aura que des routeurs nuls. Il y aura donc $(W = M \cdot K)$ routeurs en tout.

Ils seront numérotés de $k = 1,...,W$ et on supposera que les routeurs $1,...,M$ sont sur le site 1, les routeurs $M + 1,...,2M + 1$ sur le site 2, etc.

Pour simplifier on notera Ω_j l'ensemble des routeurs du site $j = 1,...,K$. De la même façon, nous supposons que chaque routeur $k = 1,...,W$ dispose de $\overline{s}$ cartes de communication. On prendra,

$$\overline{s} = \max_{r=1,...,R} \overline{s}_r \tag{6.8}$$

Evidemment, il faudra ajouter au problème des contraintes pour exprimer que si le routeur k est du type r, le nombre de cartes effectives ne devra pas dépasser s_r .

C.1.2.1. Notations

L'équipement nécessaire dépend du nombre de clients à connecter, de leurs demandes, et de la façon dont on affecte les clients aux équipements.

On introduit les notations suivantes.

- Pour $k = 1,...,W$ et $r = 0,...,R$,

$$\varepsilon_r^k = \begin{cases} 1 & si\ le\ routeur\ k\ est\ de\ modèle\ r \\ 0 & sinon \end{cases}$$

- Pour $c = 1,...,s$, $k = 1,...,W$ et $i = 1,...,N$,

$$y_{kc}^i = \begin{cases} 1 & si\ le\ client\ i\ est\ connecté\ à\ la\ carte\ c\ du\ routeur\ k \\ 0 & sinon \end{cases}$$

- Pour $c = 1,...,s$, $k = 1,...,W$ et $t = 0,...,l$,

$$z_{kc}^t = \begin{cases} 1 & si\ la\ carte\ c\ du\ routeur\ k\ est\ du\ modèle\ t \\ 0 & sinon \end{cases}$$

C.1.2.2. Contraintes

Etant donné un vecteur d'affectation $x = \left[x_{ij} \right]$ des nœuds clients aux sites, on peut énoncer les contraintes suivantes pour qu'une solution d'équipement du site $j = 1,...,K$ soit valide :

- Chaque routeur est d'un seul modèle (éventuellement nul),

$$\sum_{r=0}^{R} \varepsilon_r^k = 1 \qquad \forall k \in \Omega_j \tag{6.9}$$

- Chaque carte de communication est d'un seul modèle (éventuellement nul),

$$\sum_{t=0}^{I} z_{kc}^t = 1 \qquad \forall c = 1,...,\overline{s}; \forall k \in \Omega_j \tag{6.10}$$

- Le nombre de cartes effectives sur un routeur est inférieur au nombre de slots disponibles,

$$\sum_{c=1}^{\overline{s}} \sum_{t=1}^{I} z_{kc}^{t} \leq \sum_{r=0}^{R} \varepsilon_{r}^{k} \cdot \overline{s}_{r} \qquad \forall k \in \Omega_{j} \tag{6.11}$$

On remarquera que si le modèle de routeur retenu est le modèle nul, cette contrainte assure qu'aucune carte effective ne pourra être placée dessus.

- Chaque client affecté au site j doit être raccordé à une et une seule interface d'un routeur de ce site,

$$\sum_{k \in \Omega_{j}} \sum_{c=1}^{\overline{s}} y_{kc}^{i} = x_{ij} \qquad i = 1,...,N \tag{6.12}$$

Cette contrainte impose également que si le client i n'est pas raccordé au site j $j = 1,...,K$, alors toutes les variables sont nulles.

- Le débit de chaque port de la carte c du routeur k doit être égal à la demande de chacun des clients affectés à cette carte,

$$\sum_{t=1}^{I} z_{kc}^{t} \cdot r_{t} = y_{kc}^{i} \cdot d_{i} \qquad i = 1,...,N; c = 1,...,\overline{s}; k \in \Omega_{j} \tag{6.13}$$

On remarquera que cette contrainte garantit qu'un client ne peut pas être affecté à une carte nulle. On impose de plus qu'un client demandant un lien à 2 Mbps ne soit pas connecté sur un port à 10 Mbps.

- Le nombre de clients affectés à la carte de communication c du routeur k est inférieur au nombre de ports sur cette carte,

$$\sum_{i=1}^{N} y_{kc}^{i} \leq \sum_{t=1}^{I} z_{kc}^{t} \cdot p_{t} \qquad c = 1,...,\overline{s}; k \in \Omega_{j} \tag{6.14}$$

- La capacité du bus interne du routeur k doit être supérieure à la somme des débits crêtes des cartes placées sur ce routeur,

$$\sum_{c=1}^{\overline{s}} \sum_{t=1}^{I} z_{kc}^{t} \cdot r_{t} \cdot p_{t} \leq \sum_{r=0}^{R} \varepsilon_{r}^{k} \cdot \overline{T}_{r} \qquad k \in \Omega_{j} \tag{6.15}$$

Cette contrainte garantit également qu'un client ne peut être affecté à un routeur nul vu que le routeur nul ne peut supporter que des cartes nulles (et un client ne peut pas être affecté à une carte nulle).

C.1.2.3. Coût des équipements

Le coût global des équipements du site j s'exprime de la façon suivante,

$$E_j(x) = \sum_{k \in \Omega_j} \sum_{r=0}^{R} \varepsilon_r^k \cdot \psi_r + \sum_{k \in \Omega_j} \sum_{c=1}^{\bar{s}} \sum_{t=1}^{I} z_{kc}^t \cdot \phi_t \qquad (6.16)$$

Il représente le coût total des routeurs installés dans le site ainsi que les différentes cartes installées sur chaque routeur.

D. Approche 1 : La programmation linéaire

Une des approches classiques consiste à modéliser le problème de la conception de la topologie optimale à l'aide de la programmation linéaire en nombres entiers. De ce fait la modélisation que l'on propose est la suivante.

Nous rappelons qu'à travers cette modélisation nous cherchons à trouver la meilleure affectation primaire de N clients à k sites (à trouver) parmi K sites potentiels. Le coût à optimiser est le coût présenté dans la section C.1.1.4.

On formule ci-dessous le problème global à résoudre :

$$
\begin{cases}
\underset{x}{Min} \sum_{j=1}^{K} L_j \cdot u_j + \sum_{i=1}^{N}\sum_{j=1}^{K} c_{ij} \cdot x_{ij} + \sum_{j=1}^{K}\left(\sum_{k\in\Omega_j}\sum_{r=0}^{R} \varepsilon_r^k \cdot \psi_r + \sum_{k\in\Omega_j}\sum_{c=1}^{\overline{s}}\sum_{t=1}^{I} z_{kc}^t \cdot \phi_t \right) \\
sous\ les\ contraintes : \\
\sum_{j=1}^{K} x_{ij} = 1 \qquad i = 1,\ldots,N \\
x_{ij} \le u_j \qquad i = 1,\ldots,N;\ j = 1,\ldots,K \\
\sum_{r=0}^{R} \varepsilon_r^k = 1 \qquad k = 1,\ldots,W \\
\sum_{t=0}^{I} z_{kc}^t = 1 \qquad c = 1,\ldots,\overline{s};\ k = 1,\ldots,W \\
\sum_{c=1}^{\overline{s}}\sum_{t=1}^{I} z_{kc}^t \le \sum_{r=0}^{R} \varepsilon_r^k \cdot \overline{s}_r \qquad k = 1,\ldots,W \\
\sum_{k\in\Omega_j}\sum_{c=1}^{\overline{s}} y_{kc}^i = x_{ij} \qquad i = 1,\ldots,N;\ j = 1,\ldots,K \\
\sum_{t=1}^{I} z_{kc}^t \cdot r_t = y_{kc}^i \cdot d_i \qquad i = 1,\ldots,N;\ c = 1,\ldots,\overline{s};\ k = 1,\ldots,W \\
\sum_{i=1}^{N} y_{kc}^i \le \sum_{t=1}^{I} z_{kc}^t \cdot p_t \qquad c = 1,\ldots,\overline{s};\ k = 1,\ldots,W \\
\sum_{c=1}^{\overline{s}}\sum_{t=1}^{I} z_{kc}^t \cdot r_t \cdot p_t \le \sum_{r=0}^{R} \varepsilon_r^k \cdot \overline{T}_r \qquad k = 1,\ldots,W \\
x_{ij} \in \{0,1\} \qquad i = 1,\ldots,N;\ j = 1,\ldots,K \\
u_j \in \{0,1\} \qquad j = 1,\ldots,K \\
\varepsilon_r^k \in \{0,1\} \qquad r = 0,\ldots,R;\ k = 1,\ldots,W \\
y_{kc}^i \in \{0,1\} \qquad c = 1,\ldots,\overline{s};\ k = 1,\ldots,W;\ i = 1,\ldots,N \\
z_{kc}^t \in \{0,1\} \qquad t = 0,\ldots,I;\ c = 1,\ldots,\overline{s};\ k = 1,\ldots,W
\end{cases}
\tag{6.17}
$$

Il va de soi que cette modélisation n'est pas exploitable sur des problèmes de tailles réelles. En effet la complexité de celle-ci croît de manière exponentielle. Les temps d'exécution que nous avons observés deviennent rapidement prohibitoires.

Dans ce qui suit nous proposons une décomposition de ce problème basée sur l'étude des configurations optimales. Nous proposons plusieurs algorithmes (exacts, heuristiques) qui exploitent cette décomposition.

Nous présentons en fin de chapitre les résultats obtenues sur des topologies tests par ces différents algorithmes.

E. Décomposition du problème

Pour décomposer le problème précédent, nous faisons deux remarques :

- Les demandes d_i des clients $i = 1,...,N$ ne peuvent prendre que quelques valeurs et correspondent aux débits des ports de communication des interfaces. Typiquement, les valeurs de ces demandes seront : 2 Mbps, 10 Mbps, 100 Mbps ou 1 Gbps.

- Il existe un nombre relativement restreint de configurations au niveau d'un site. Notons H ce nombre. Chacune de ces configurations est caractérisée par le nombre n_t clients ayant le débit $(d_i = r_t)$ pour chaque technologie $t = 1,...,I$, et par son coût E_h.

Une configuration est donc décrite par un vecteur $(n_1,...,n_I)$ où n_t est le nombre de clients ayant le débit r_t pouvant être supporté. On remarquera que n_t est nécessairement un multiple de $p_t : \exists s_t;\ n_t = s_t \cdot p_t$. Le nombre s_t représente le nombre de cartes de type t à placer sur les routeurs du site.

Si le nombre I de modèles de cartes et le nombre R de modèles de routeurs sont assez « petits », on peut décomposer le problème en résolvant d'abord le problème des configurations optimales en fonction des vecteurs de demandes $(n_1,...,n_I)$, et ensuite le problème de l'affectation des clients aux sites.

E.1. Reformulation du problème

On rappelle que Φ_t est l'ensemble des clients $i = 1,...,N$ tels que $(d_i = r_t)$. On notera de plus $\delta_j^h = 1$ si la configuration $h = 0,...,H$ est retenue pour le site j, et 0 sinon. La configuration $(h = 0)$ représente une configuration de coût nul et de capacité nulle. Le problème peut alors être formulé de la façon suivante:

$$\begin{cases} Min_x \sum_{i,j} c_{ij} \cdot x_{ij} + \sum_{j=1}^{K} \sum_{h=1}^{H} \delta_j^h \cdot (E_h + L_j) \\ sous\ les\ contraintes : \\ \sum_{j=1}^{K} x_{ij} = 1 \qquad i = 1,...,N \\ \sum_{h=0}^{H} \delta_j^h = 1 \qquad j = 1,...,K \\ \sum_{i \in \Phi_t} x_{ij} \le \sum_{h=1}^{H} n_t^h \cdot \delta_j^h \qquad t = 1,...,I;\ j = 1,...,K \\ x_{ij} \in \{0,1\} \qquad i = 1,...,N;\ j = 1,...,K \\ \delta_j^h \in \{0,1\} \qquad h = 0,...,H;\ j = 1,...,K \end{cases} \tag{6.18}$$

Dans la section suivante, nous examinons comment calculer le coût optimal de chaque configuration. Nous examinerons ensuite comment résoudre le problème ci-dessus.

E.2. Résolution des configurations

E.2.1. Position du problème

On définit ici une configuration comme un vecteur $s = (s_1,...,s_I)$ où s_t représente le nombre de cartes du type t à placer. Une telle configuration permettra de supporter l'ensemble des vecteurs de demandes $(n_1,...,n_I)$, où $n_t \le s_t \cdot p_t$ est le nombre de clients du type $t = 1,...,I$ à connecter. Rappelons que Φ_t est l'ensemble des clients $i = 1,...,N$ tels que $(d_i = r_t)$. Le nombre maximal de cartes du type t dans une configuration est donc $h_t = |\Phi_t| / p_t$.

L'ensemble des configurations possibles est donné par,

$$\Lambda = \{ s \mid \ 0 \le s_t \le h_t \ ; \ t = 1,...,I \} \tag{6.19}$$

On note $H = |\Lambda|$,

$$H = \prod_{t=1}^{I} (h_t + 1) \tag{6.20}$$

On remarquera que cet ensemble de configurations contient la configuration $(h_1,...,h_I)$ qui permet de supporter l'affectation centralisée de l'ensemble des clients à un site unique.

Pour simplifier l'implémentation informatique, on peut numéroter les configurations de la façon suivante :

$$id(s_1,...,s_I) = \sum_{t=1}^{I} s_t \cdot \prod_{u=t+1}^{I} (h_u + 1) \qquad \forall s \in \Lambda \tag{6.21}$$

Avec cette numérotation, la configuration nulle a le numéro *0* et la configuration maximale a le numéro $(H-1)$.

Notre objectif est de construire la fonction $\mu^* : \Lambda \to \mathbb{R}$ qui à chaque configuration s associe le coût optimal des équipements permettant de supporter cette configuration.

Ce coût optimal peut se décomposer en d'une part le coût des cartes et d'autre part le coût des routeurs :

$$\mu^*(s) = \mu_c^*(s) + \mu_r^*(s) \qquad \forall s \in \Lambda \tag{6.22}$$

E.2.2. Problème d'optimisation en nombres entiers

Considérons d'abord le coût des cartes. Il est donné par,

$$\mu_c^*(s) = \sum_{t=1}^{I} s_t \cdot \phi_t \tag{6.23}$$

Considérons maintenant le coût des routeurs. On notera que si $(s' \geq s)$ alors on a nécessairement $\left(\mu_r^*(s') \geq \mu_r^*(s)\right)$ puisque tout ensemble de routeurs supportant la configuration s' doit supporter la configuration s.

Il reste à déterminer le coût des routeurs permettant de supporter s_1 cartes du type 1, s_2 cartes du type 2, etc. Pour cela, on suppose que l'on dispose de M routeurs de chaque modèle avec M assez grand pour que cela ne pose pas de problème. On note $r(j)$ le modèle du routeur $j = 1,...,(R \cdot M)$. Mathématiquement, le problème peut s'écrire,

$$\begin{cases} \mu_r^*(s) = Min_{x,\varepsilon} \sum_{j=1}^{R \cdot M} \varepsilon_j \cdot \psi_{r(j)} \\ \textit{sous les contraintes :} \\ \sum_{j=1}^{R \cdot M} x_{jt} = s_t \qquad t = 1,...,I \\ \sum_{t=1}^{I} x_{jt} \leq \varepsilon_j \cdot \overline{s}_{r(j)} \qquad j = 1,...,R \cdot M \\ \sum_{t=1}^{I} x_{jt} \cdot r_t \cdot p_t \leq \varepsilon_j \cdot \overline{T}_{r(j)} \qquad j = 1,...,R \cdot M \\ \varepsilon_j \in \{0,1\} \qquad j = 1,...,R \cdot M \\ x_{jt} \in \mathbb{N} \qquad t = 1,...,I \ ; \ j = 1,...,R \cdot M \end{cases} \tag{6.24}$$

Dans ce problème, les variables ε_j sont à 1 si le routeur j est utilisé et 0 sinon. Les variables x_{jt} représentent le nombre de cartes du type t placées sur le routeur j. La première contrainte exprime la conservation du nombre de cartes. Les deux suivantes expriment les limitations de chaque routeur utilisé en nombre de slots et en capacité de la fabrique.

Formulé ainsi, la construction de la fonction $\mu_r^*(s)$ requiert la résolution de H fois le problème précédent pour chaque configuration $(s \in \Lambda)$. Nous verrons qu'il est possible d'utiliser les relations entre ces différents problèmes pour les résoudre de manière très efficace.

E.2.3. Formulation en programmation dynamique

Considérons une configuration s et notons $n = \sum_t s_t$ le nombre total de cartes à placer. Nous formulons ci-dessous le problème de résolution du coût optimal $\mu_r^*(s)$

comme un problème de décisions séquentielles. Clairement ce coût optimal ne dépend pas de l'ordre dans lequel on considère les cartes. On verra qu'il est avantageux de les considérer par ordre de débit croissant. Pour chaque carte $k = 1,...,N$, on note $t(k)$ le type de cette carte.

E.2.4. Etat du système

On considère le placement des cartes sur les $(M \cdot R)$ routeurs les unes après les autres. A l'étape k, après avoir déjà placé $(k-1)$ cartes, le système est dans l'état x^k. Cet état du système est alors décrit par l'ensemble $A^k(x^k)$ des routeurs actifs, i.e. auxquels on a affecté au moins une des $(k-1)$ cartes. Pour chaque routeur j, on dispose de plus de sa bande-passante résiduelle $T_j^{res}(x^k)$ et de son nombre de slots résiduels $s_j^{res}(x^k)$.

E.2.5. Commandes

Pour la prochaine carte k il faut prendre une décision de placement u_k. On considère deux types de placement :

- Placement sur un routeur déjà actif capable d'accueillir cette carte,
- Placement sur un nouveau routeur.

On définit donc l'ensemble $B^k(x^k) \subset A^k(x^k)$ des routeurs actifs vérifiant les deux conditions suivantes :

$$s_j^{res}(x^k) \geq 1 \quad et \quad T_j^{res}(x^k) \geq r_{t(k)} \cdot p_{t(k)} \tag{6.25}$$

La première condition impose qu'un routeur n'appartient à $B^k(x^k)$ que si il lui reste au moins un slot de libre et la deuxième impose qu'il lui reste assez de bande passante pour accueillir la carte k.

Enfin, on définit l'ensemble $I^k(x^k)$ de routeurs de la façon suivante,

$$I^k(x^k) = \left\{c_1^k(x^k),...,c_R^k(x^k)\right\} \quad où \quad c_r^k(x^k) = \min_{j \notin A^k(x^k)} \left\{j : r(j) = r\right\} \tag{6.26}$$

$I^k(x^k)$ représente l'ensemble des routeurs non encore actifs qui sont candidats à être utilisés. Il ne sert en effet à rien de considérer l'ensemble des routeurs non utilisés : il suffit pour chaque décision d'en considérer un de chaque modèle.

A l'étape k, dans un état x^k, l'ensemble des décisions u^k de placement pour la carte k est donné par $C^k(x^k) = B^k(x^k) \cup I^k(x^k)$. Autrement dit on ne considère que les

routeurs actifs qui satisfont aux contraintes sur le nombre de ports et sur la bande-passante restante, et un routeur inactif de chaque catégorie.

E.2.6. Dynamique du système

Dans l'état x^k, si la commande est $u^k \in C^k\left(x^k\right)$, alors le système passe dans l'état $x^{k+1} = f\left(x^k, u^k\right)$ décrit d'une part par l'ensemble des routeurs actifs,

$$A^{k+1}\left(x^{k+1}\right) = \begin{cases} A^k\left(x^k\right) & si\ u^k \in B^k\left(x^k\right) \\ A^k\left(x^k\right) \cup u^k & si\ u^k \in I^k\left(x^k\right) \end{cases} \tag{6.27}$$

et d'autre part le nombre de slots et la bande-passante restante de chaque routeur:

$$s_{k+1}^{res}\left(x^{k+1}\right) = s_k^{res}\left(x^k\right) - 1\ \ et\ \ T_{k+1}^{res}\left(x^{k+1}\right) = T_k^{res}\left(x^k\right) - r_{t(k)} \cdot p_{t(k)} \tag{6.28}$$

les valeurs des autres routeurs $\left(j \neq u^k\right)$ restant inchangées.

E.2.7. Coûts des décisions

Notons $g_k\left(x^k, u^k\right)$ le coût de la décision u^k pour placer la carte k dans l'état x^k :

$$g_k\left(x^k, u^k\right) = \begin{cases} \psi_{r(u^k)} & si\ \ u^k \in I^k\left(x^k\right) \\ 0 & sinon \end{cases} \tag{6.29}$$

Le problème à résoudre consiste alors à trouver la séquence optimale $\left(u_1, ..., u_n\right)$ de décisions permettant de minimiser le coût des routeurs,

$$\mu_r^*(s) = Min_{u_1,...,u_n} \sum_{k=1}^{n} g_k\left(x^k, u^k\right) \tag{6.30}$$

partant de l'état initial $x^0 = [0]$ et en suivant la dynamique $x^{k+1} = f\left(x^k, u^k\right)$.

Pour déterminer la séquence optimale des décisions $\left(u_1, ..., u_n\right)$ nous utilisons l'algorithme 1 qui est de type Branch and Bound. Cet algorithme prend en paramètre la configuration, la carte k à placer, l'état x^k atteint après avoir placées les $(k-1)$ premières cartes ainsi que le coût de cette solution partielle.

Les derniers paramètres sont une borne supérieure sur le coût, une borne inférieure sur le coût et une variable booléenne stop indiquant que la borne inférieure a été atteinte.

Algorithm 1 Algorithme d'optimisation des configurations

```
1: procedure BranchAndBound(k, x^k, cost, ub, lb, stop)
2:     if k=n+1 then                          ▷ Toutes les cartes sont placées
3:         Mettre à jour ub si cost < ub et faire stop=true si cost = lb
4:         return 0
5:     end if
6:     Calculer B^k(x^k) et I^k(x^k)
7:     bestCost = 0
8:     for u_k ∈ B^k(x^k) ∪ I^k(x^k) do
9:         costToGo = g_k(x^k, u_k)
10:        if cost + costToGo < ub then
11:            costToGo = costToGo + BranchAndBound(k + 1, f(x^k, u_k),
                   cost+costToGo, ub, lb, stop)
12:            if costToGo < bestCost then
13:                bestCost = costToGo
14:            end if
15:            if stop then
16:                return bestCost
17:            end if
18:        end if
19:    end for
20: end procedure
```

Figure 62: PSeudo code de l'Algorithme 1

Diverses optimisations sont possibles en utilisant les relations existantes entre les différentes configurations.

E.2.8. Utilisation des configurations précédentes

Il est possible d'éviter d'utiliser l'algorithme précédent pour déterminer le coût optimal d'une configuration $(s \neq 0)$. En effet, à cette configuration on peut toujours associer une configuration s' qui ne diffère de s que parce qu'elle a une carte en moins. Par exemple, à la configuration $s = (1,0,3)$ on associera la configuration $s' = (1,0,2)$ et à la configuration $s = (2,0,0)$ on associera la configuration $s' = (1,0,0)$. Nous disons que la configuration s' est la plus proche de s.

Etant donné l'ordre dans lequel nous considérons les configurations, la configuration s' a toujours été résolue quand on doit résoudre la configuration s. On connaît donc son coût optimal $\mu_r^*(s')$

Supposons qu'à chaque fois que l'on résout une configuration, on garde une trace de la bande-passante résiduelle maximale sur les routeurs actifs disposant d'au moins un slot libre. Supposons également que parmi les routeurs actifs disposant de cette bande-passante résiduelle maximale, on conserve le nombre maximal de slots libres.

Quand on résout la configuration s, si la configuration s' a assez de bande-passante résiduelle pour accueillir la nouvelle carte, alors on sait immédiatement que $\mu_r^*(s) = \mu_r^*(s')$.

Ce principe permet donc de résoudre directement de nombreuses configurations directement en utilisant les configurations précédemment résolues.

L'information conservée étant partielle, il est possible que ce test heuristique déclare que la configuration s ne peut être supportée en utilisant les mêmes routeurs que pour la configuration s', alors que pourtant c'est le cas. Si cela arrive, la résolution se fera avec la méthode du Branch and Bound, mais avec une borne inférieure qui sera exacte.

E.2.9. Borne inférieure

Une borne inférieure efficace peut être obtenue en utilisant le même principe. Clairement, $\mu_r^*(s')$ est une borne inférieure sur le coût optimal de la configuration s. Cette borne inférieure sera atteignable si avec les routeurs de la configuration s' il est possible de supporter la configuration s. Si cette borne inférieure est atteinte au cours du Branch and Bound, on peut immédiatement arrêter l'exploration: le coût optimal a été atteint.

E.2.10. Borne supérieure

La configuration s ne différant de la configuration s' que par une seule carte, il est clair qu'il est possible de supporter la configuration s en ajoutant aux équipements permettant de supporter la configuration s' un nouveau routeur auquel on affectera la nouvelle carte. Une borne supérieure sur $\mu_r^*(s)$ est donc obtenue en ajoutant à $\mu_r^*(s')$ le coût minimal d'un routeur supportant la nouvelle carte.

Comme décrit sur l'algorithme 1, cette borne supérieure peut être mise à jour au cours de l'exploration des solutions. Elle va permettre d'élaguer l'énumération des solutions.

E.2.11. Utilisation du principe d'optimalité

L'utilisation du principe d'optimalité de Bellman permet d'améliorer significativement le calcul du coût optimal des équipements d'une configuration.

Considérons une configuration s. Supposons que lors du placement de la carte k, tous les routeurs actifs ne puisse accepter cette carte, i.e. $A^k(x^k) = \varnothing$. Les cartes étant considérées par ordre de débit croissant, cela signifie que les routeurs sur lesquels on a placées les $(k-1)$ premières cartes n'accepteront plus aucune autre carte. Les cartes restantes $k,...,n$ devront donc être placées sur de nouveaux routeurs.

En supprimant ces $(k-1)$ premières cartes de la configuration s, on obtient une configuration $\tilde{s}$ qui a nécessairement été déjà résolue. De par le principe d'optimalité, le coût minimal pour placer les cartes $k,...,n$ est donc égal à $\mu_r^*(\tilde{s})$. Il est donc inutile de continuer l'exploration récursive des solutions.

Suivant le coût global (coût de la solution partielle plus coût minimal à venir), on mettra à jour la borne supérieure et éventuellement, si la borne inférieure est atteinte, on arrêtera l'algorithme.

E.2.12. Un exemple

Pour illustrer l'efficacité de l'algorithme (exact) précédent, prenons un exemple simple.

Nous considérons 3 modèles de carte:

- Modèle 1: $p_1 = 10\ ports,\ r_1 = 2Mbps,\ \phi_1 = 3Keuros,$
- Modèle 2: $p_2 = 10\ ports,\ r_2 = 10Mbps,\ \phi_2 = 10Keuros,$
- Modèle 3: $p_3 = 10\ ports,\ r_3 = 100Mbps,\ \phi_3 = 80Keuros.$

On suppose qu'il y a 100 clients demandant un lien à 2 Mbps, 50 demandant un lien à 10 Mbps et 30 voulant un lien à 100 Mbps. On a alors $(h_1 = 10,\ h_2 = 5,\ h_3 = 3)$. L'ensemble des configurations possibles est donné par,

$$\Lambda = \left\{ s \,\middle|\, 0 \le s_1 \le 10,\ 0 \le s_2 \le 5,\ 0 \le s_3 \le 3 \right\}$$

et le nombre total de configurations est $H = 11 \times 6 \times 4 = 264$. La configuration *0* est la configuration nulle $(0,0,0)$ et la configuration *263* est la configuration centralisée $(10,5,3)$.

Pour supporter chacune de ces configurations, on dispose de 2 modèles de routeurs:

- Modèle 1: $\overline{s}_1 = 15\ slots,\ \overline{T}_1 = 150Mbps,\ \psi_1 = 100Keuros,$
- Modèle 2: $\overline{s}_2 = 8\ slots,\ \overline{T}_2 = 300Mbps,\ \psi_2 = 143Keuros.$

Pour construire la fonction $\mu^* : \Lambda \to \aleph$ qui à chaque configuration s associe le coût optimal des équipements permettant de supporter cette configuration, il faut résoudre chacune des 264 configurations.

Sur les 263 configurations non nulles, 115 peuvent être résolues sans utiliser le Branch and Bound, soit environ 44 %. Sur les 148 configurations restantes, la borne inférieure est atteinte 13 fois, soit environ 9% des fois. Au total, les 264 configurations sont résolues en 0.15 secondes.

F. Relaxation Lagrangienne

Nous réutilisons la formulation (6.18) du problème et nous relaxons la contrainte de couplage :

$$\sum_{i\in\Phi_t} x_{ij} \le \sum_{h=0}^{H} n_t^h \cdot \delta_h^j \qquad t = 1,...,I;\ j = 1,...,K \tag{6.31}$$

Le lagrangien de la fonction coût s'écrit donc :

$$L(x,\delta,\lambda) = \sum_{ij} c_{ij} \cdot x_{ij} + \sum_{j=1}^{K}\sum_{h=1}^{H} \delta_j^h \cdot \left(E_h + L_j\right) + \sum_{j=1}^{K}\sum_{t=1}^{I} \lambda_j^t \cdot \left(\sum_{i\in\Phi_t} x_{ij} - \sum_{h=1}^{H} n_t^h \cdot \delta_j^h\right) \tag{6.32}$$

que l'on peut réécrire sous la forme :

$$L(x,\delta,\lambda) = \sum_{j=1}^{K}\sum_{t=1}^{I}\sum_{i\in\Phi_t} x_{ij} \cdot \left(c_{ij} + \lambda_j^t\right) + \sum_{j=1}^{K}\sum_{h=1}^{H} \delta_j^h \cdot \left(E_h + L_j - \sum_{t=1}^{I} \lambda_j^t \cdot n_t^h\right) \tag{6.33}$$

pour $\lambda = \left(\lambda_j^t\right)$ fixé, on définit la fonction duale

$$W(\lambda) = \underset{x,\delta}{Min}\, L(x,\delta,\lambda) = \underset{x}{Min} \sum_{j=1}^{K}\sum_{t=1}^{I}\sum_{i\in\Phi_t} x_{ij} \cdot \left(c_{ij} + \lambda_j^t\right) + \underset{\delta}{Min} \sum_{j=1}^{K}\sum_{h=1}^{H} \delta_j^h \cdot \left(E_h + L_j - \sum_{t=1}^{I} \lambda_j^t \cdot n_t^h\right) \tag{6.34}$$

On doit donc résoudre deux problèmes indépendants. Le premier s'écrit :

$$\begin{cases} \underset{x}{Min} \sum_{j=1}^{K}\sum_{t=1}^{I}\sum_{i\in\Phi_t} x_{ij} \cdot \left(c_{ij} + \lambda_j^t\right) \\ sous \\ \sum_{j=1}^{K} x_{ij} = 1 \qquad i = 1,...,N \\ x_{ij} \in \{0,1\} \qquad i = 1,...,N,\ j = 1,...,K \end{cases} \tag{6.35}$$

pour $i \in \Phi_t$ fixé la solution optimale est de prendre $x_{ij}^* = 1$ pour $j^* = \underset{j}{argmin}\left(c_{ij} + \lambda_j^t\right)$ et $x_{ij} = 0$ pour $j \neq j^*$.

Le second problème s'écrit :

$$\begin{cases} \underset{\delta}{Min} \sum_{j=1}^{K} \sum_{h=1}^{H} \delta_j^h \cdot \left(E_h + L_j - \sum_{t=1}^{I} \lambda_j^t \cdot n_t^h \right) \\ sous \\ \sum_{h=1}^{H} \delta_j^h = 1 \qquad j = 1,...,K \\ \delta_j^h \in \{0,1\} \qquad h = 0,...,H, \;\; j = 1,...,K \end{cases} \tag{6.36}$$

Ce problème est séparable sur le sites j. Pour un site j donnée, on a deux cas :

- $E_h + L_j - \sum_{t=1}^{I} \lambda_j^t \cdot n_t^h > 0 \;\; \forall h \geq 1$ alors la solution optimale est
$$\delta_0^j = 1 \text{ et } \delta_h^j = 0 \;\; \forall h \neq 0$$
- Sinon, la solution optimale est
$$\delta_{h^*}^j = 1, \text{ où } h^* = \underset{h}{\operatorname{argmin}} \left(E_h + L_j - \sum_{t=1}^{I} \lambda_j^t \cdot n_t^h \right) \text{ et } \delta_h^j = 0 \;\; \forall h \neq h^*$$

Finalement le problème à résoudre est :

$$\underset{\lambda}{Max} \{ W(\lambda) : \lambda \geq 0 \} \tag{6.37}$$

on va utiliser pour cela l'algorithme dual basé sur la relaxation Lagrangienne, les dérivées en λ_p^r sont données par les formules suivantes :

$$\frac{\partial W}{\partial \lambda_p^r} = \sum_{i \in \Phi_t} x_{ip}^* - \sum_{h=1}^{H} n_r^h \cdot \delta_h^{p*} \quad r = 1,...,I; \; p = 1,...,K \tag{6.38}$$

ou encore

$$\frac{\partial W}{\partial \lambda_p^r} = \sum_{i \in \Phi_t} x_{ip}^* - n_r^{h^*} \quad r = 1,...,I; \; p = 1,...,K \tag{6.39}$$

Cette approche bien que prometteuse est loin d'être satisfaisante pour la résolution du problème 6.18. Elle permet néanmoins d'obtenir une « bonne » borne inférieure de ce problème en résolvant l'approche duale. Elle permet aussi d'obtenir une « bonne » borne supérieure par la résolution du problème primal. Celle ci est inférée par la solution du dual (solution en x) et permet donc une fois que les affectations sont fixés de résoudre le problème des configurations optimales sur chaque site.

Nous exploitons les bornes inférieures et supérieures pour la résolution de l'approche exacte présenté précédemment. Cela permet d'améliorer efficacement le Branch and Bound.

G. Approche 2 : Heuristique SearchCutExplore

L'analyse des coûts présentée dans la section C.1. nous à permis de concevoir une heuristique qui exploite cette dualité entre les deux types de coûts (ceux qui poussent à la centralisation et ceux qui poussent vers une solution distribuée).

L'idée de base de cette heuristique exploite une technique de « clustering » en plus de la dualité précédemment évoquée.

En effet l'heuristique est un processus récursif qui à chaque niveau et grâce à un algorithme performant de « clustering » [KAN02] sépare le problème en deux régions (gauche et droite) et calcule le coût de la solution centralisée.

Pour ce faire on cherche le site le plus proche du barycentre de la région et on affecte tous les clients du niveau à ce site. On calcule à partir de cette affectation le coût des liens et des équipements pour cette solution. Sans oublier d'ajouter le coût du site choisi.

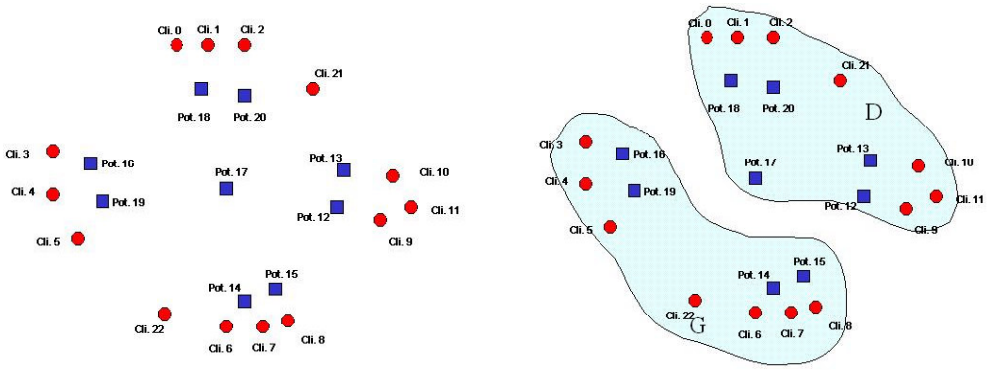


Figure 63: Première itération de l'heuristique searchCutExplore

Une fois que le processus récursif atteint la profondeur maximale (plus qu'un site dans la région), celui ci compare à chaque niveau la solution centralisée à la somme des coûts des solutions gauche et droite. Il retient le meilleur coût et par la même occasion l'affectation associée.

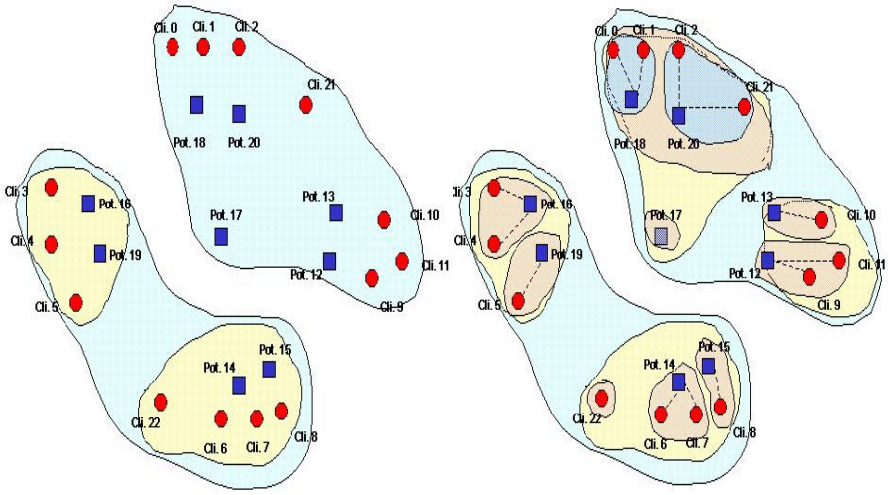


Figure 64: Itération intermédiaire et profondeur maximale de la récursivité

Ce processus est entièrement guidé par la position géographique des sites et des clients à un point près. A chaque itération nous remettons en cause les affectations gauche droite des clients. Ce procédé est effectué selon une technique de recherche locale qui explore un voisinage de type « meilleur réaffectation » pour chaque client. Ainsi nous cherchons toutes les réaffectations possibles qui font diminuer le coût global.

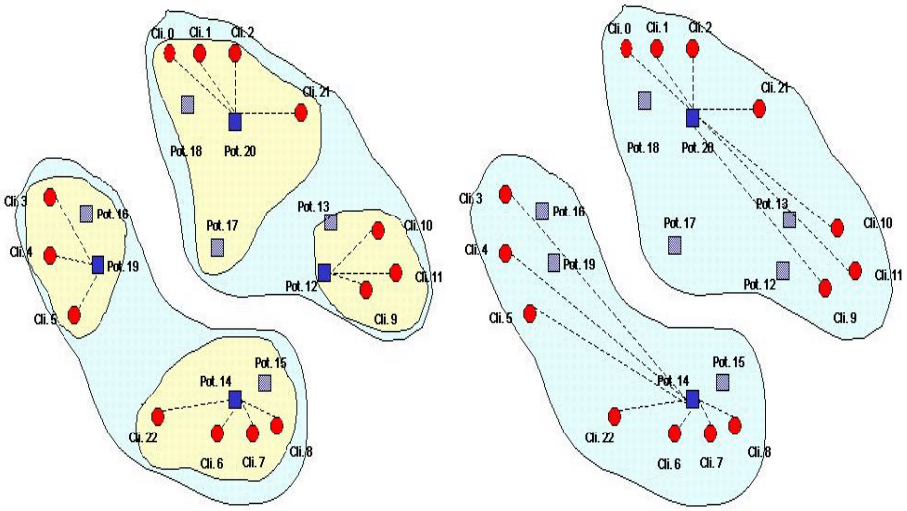


Figure 65: Calcul de l'affectation par comparaison du coût centralisé et distribué

Le calcul du coût des sites se base sur la résolution des configurations optimales présentées précédemment. En effet une fois le problème des configurations résolu il suffit de fournir la liste des clients affectés au site pour obtenir la configuration optimale « la plus proche » qui supporte cette demande.

```
fusion (clientsLeft, sitesLeft, clientsRight, sitesRight, kmeanCost)
{
  cost = kmeanCost;
  currentCost = 0.0;

  // Calculer les meilleurs améliorations que l'on
  // peut apporter au voisinage courant
  tant que ( currentCost < cost )
  {
    // Effectuer la meilleure modification locale
    currentCost  = bestMove(clientsLeft, sitesLeft, clientsRight,
                            sitesRight, bestClient, bestSite, cost);
    // Enregistrer la modification
    solution[bestClient] = idBestSite ;

    // Mettre à jour le coût courant
    If ( currentCost < cost )
      cost = curentCost;
  }
  return cost;
}
```

Figure 66: Pseudo code : exploration du voisinage

bestMove teste toutes les affectations des nœuds gauche sur les sites de droites et vice versa. Elle garde la meilleure affectation qui fait diminuer le coût de la solution.

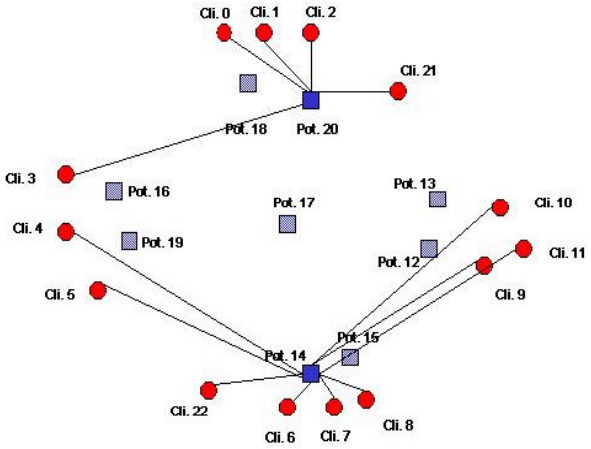


Figure 67: Optimisation des affectations grâce à la procédure bestMove

Le gain obtenu dans ce cas en changeant les affectations des clients 9,10,11 du site 20 vers le site 14 et du client 3 vers le site 20 est de l'ordre de 2220.33 pour un coût total de 67453.7 soit une gain de 3.29%.

```
searchCutExplore( Clients , Sites )
{
   // Si pas de sites, on retourne un coût infini: ce n'est pas une solution
  if ( Sites.empty() ) return INFTY;

  // Si pas de clients, on retourne 0
  if ( Clients.empty() ) return 0;

  // Recherche de la meilleure solution centralisée
  bestCost = searchBestAll(Clients, Sites);

  // Si on a au moins 2 sites et au moins 2 nœuds,
  // on explore récursivement en découpant en 2 régions
  if ( (Sites.size() > 1) && (Clients.size()>1)  )
    {
      // Calcul des 2 régions
      Kmean( 2, Clients, Sites);

     // Affectation des clients et des sites aux régions gauche/droite
     Affectation(Clients,Sites, leftClients, leftSites,
                 rightClients, rightSites) ;

      // Exploration de la région gauche
      kmeanCostLeft  = searchCutExplore(leftClients, leftSites);

      // Exploration de la région droite
      kmeanCostRight = searchCutExplore(rightNodes, rightSites);
    }

  // Faire la somme pour obtenir le coût le solution globale
  kmeanCost = kmeanCostLeft + kmeanCostRight;

  // Si le coût de solution centralisée est plus petit que celui de la
  // solution distribuée
  if ( bestCost <= kmeanCost ){
      // Positionner la solution Centralisée
      cost = bestCost;}
  // Sinon
  else{ cost = kmeanCost;}

 // Amélioration de la solution courante par exploration du voisinage
 if ( leftSites.size() && rightSites.size() ){
      localSearchCost = fusion ( leftClients, leftSites, rightClients,
                                 rightSites, kmeanCost);

       if (localSearchCost < cost)    cost = localSearchCost;
    }
    return cost;
}
```

Figure 68: Pseudo code de la fonction searchCutExplore

Nous avons testé cette heuristique sur des problèmes de tailles assez grande. Les temps d'exécution sont très rapides. La faible complexité de l'heuristique laisse présager des temps d'exécution pas trop élevés sur des réseaux de tailles réelles.

Nous utilisons aussi cette heuristique pour fournir une borne supérieure pour l'approche exacte.

H. Approche exacte

Pour résoudre la forme décomposé du problème initial (6.18) nous proposons un algorithme de résolution exacte. Ce dernier s'appuie sur la programmation dynamique. Nous reprenons les mêmes notations que précédemment.

Supposons que les $i-1$ premiers clients ont été affectés à des sites. Il faut affecter le client i.

H.1. Etat du système

L'état x^i du système est donné par le triplet $\left(u^i, n^i, s^i\right)$ où,

- Le vecteur $u^i = \left[u_1^i, ..., u_K^i\right]$ représente l'utilisation des sites, ($u_j^i = 1$ si le site est ouvert, 0 sinon).
- La matrice $n^i = \left[n_{j,t}^i\right]$ donne pour chaque site $j = 1, ..., K$ et chaque technologie $t = 1, ..., I$ le nombre $n_{j,t}^i$ de clients ayant la technologie t affectés au site j.
- Le vecteur $s^i = \left[s_1^i, ..., s_K^i\right]$ donne la configuration $s_j^i \in \Lambda$ de moindre coût sur chaque site $j = 1, ..., K$ permettant de supporter le vecteur de demandes $\left[n_{j,1}^i, ..., n_{j,I}^i\right]$.

H.2. Commandes

Pour le prochain client i il faut prendre une décision d'affectation à un site. Notons a_i le site choisi. On a $a_i \in \{1, ..., K\}$.

H.3. Dynamique du système

Dans l'état $x^i = \left(u^i, n^i, s^i\right)$, si la décision d'affectation $a_i \in \{1, ..., K\}$ est prise, le système évolue vers l'état $x^{i+1} = f\left(x^i, a_i\right)$ donné par les relations suivantes :

- Utilisation des sites,

$$u_j^{i+1} = \begin{cases} 1 & si\ j = a_i \\ u_j^i & sinon \end{cases} \tag{6.40}$$

- Nombre de clients par technologie sur chaque site,

$$n_{j,t}^{i+1} = \begin{cases} n_{j,t}^i + 1 & si\ j = a_i\ et\ j \in \Phi_t \\ n_{j,t}^i & sinon \end{cases} \tag{6.41}$$

- Configuration des sites,

$$s_j^{i+1} = \begin{cases} argmin_{s \in \Lambda} \left\{\mu^*(s) : s_t \cdot p_t \geq n_{j,t}^{i+1}; t = 1, ..., I\right\} & si\ j = a_i \\ s_j^i & sinon \end{cases} \tag{6.42}$$

H.4. Coût des décisions

Notons $g_i\left(x^i, a_i\right)$ le coût de la décision a_i dans l'état x^i. En notant $j = a_i$, on a :

$$g_i\left(x^i, j\right) = c_{ij} + \left(u_j^{i+1} - u_j^i\right) \cdot L_j + \mu^*\left(s_j^{i+1}\right) - \mu^*\left(s_j^i\right) \tag{6.43}$$

Le problème à résoudre consiste à déterminer la séquence optimale d'affectations $\left(a_1^*, ..., a_N^*\right)$ permettant de minimiser le coût global :

$$J_N\left(a_1^*, ..., a_N^*\right) = \min_{a_1, ..., a_N} J_N\left(a_1, ..., a_N\right) \tag{6.44}$$

où,

$$J_N\left(a_1, ..., a_N\right) = \sum_{i=1}^{N} g_i\left(x^i, a_i\right) \tag{6.45}$$

l'état initial x^0 étant tel que $u^0 = 0, n^0 = 0, s^0 = 0$.

H.5. Principe de résolution : Branch and Cut

Afin de résoudre la modélisation précédente nous utilisons la technique du « Branch and Cut ». Nous améliorons les performances de cette dernière en terme de temps de calcul en utilisant les techniques suivantes.

H.6. Une borne supérieure

En effet nous utilisons l'heuristique searchCutExplore pour fournir une borne supérieure globale de très bonne qualité. Celle ci est améliorée à chaque fois qu'une meilleure solution est découverte. Cela permet d'accélérer la méthode surtout quand la borne supérieure devient proche de l'optimum.

H.7. Une borne inférieure

La borne inférieure est locale, c'est à dire qu'a chaque sous problème (i clients affectés $N - i$ à placer) que l'on doit résoudre nous calculons une borne inférieure.

L'analyse du coût de la solution c.f. (6.17) permet de dire qu'une bonne borne inférieure est obtenue en additionnant les coût suivants :

- Coût des sites ouverts,
- Coût des distances de la solution partielle pour les clients affectés,
- Coût des distances au plus près pour les clients à affecter,

- Coût des routeurs de la solution partielle,
- Coût optimal des cartes pour supporter l'ensemble des clients.

Cette borne inférieure à l'avantage de pouvoir être calculée simplement.

H.8. Des coupes sur les distances

Soit X l'ensemble des affectations des clients aux sites admissibles, c'est à dire des affectations x vérifiant :

$$\begin{aligned} &\sum_j x_{ij} = 1 \quad i = 1,...,N \\ &x_{ij} \in \{0,1\} \end{aligned} \tag{6.46}$$

Pour une affectation $x \in X$, notons $E_j(x)$ le coût optimal du site $j = 1,...,K$ (incluant le coût d'ouverture) c'est à dire :

$$\begin{cases} E_j(x) = min_{\delta^j} \sum_{j=1}^{K} \sum_{h=1}^{H} \delta_h^j \cdot (E_h + L_j) \\ sous\ les\ contraintes \\ \sum_{h=0}^{H} \delta_h^j = 1 \qquad j = 1,...,K \\ \sum_{i \in \Phi_t} x_{ij} \le \sum_{h=1}^{H} n_t^h \cdot \delta_j^h \qquad t = 1,...,I;\ j = 1,...,K \\ \delta_h^j \in \{0,1\} \qquad h = 1,...,H;\ j = 1,...,K \end{cases} \tag{6.47}$$

Notons que le coût $E_j(x)$ existe et est fini pour tout site $j = 1,...,K$ quelle que soit l'affectation $x \in X$.

Le problème à résoudre peut être écrit :

$$\begin{cases} \Gamma^* = min_x \sum_{i,j} x_{ij} \cdot c_{ij} + \sum_{j=1}^{K} E_j(x) \\ sous\ les\ contraintes \\ x \in X \end{cases} \tag{6.48}$$

on noteras $\Gamma(x)$ le coût d'une solution admissible $x \in X$. De plus on notera X_i^k l'ensemble des solutions $x \in X$ telle que le client i est affecté au site k.

Nous allons utiliser la proposition suivante.

Proposition 1 :

Pour toute technologie $t \in \{1,...,I\}$, *tout client* i *et* i' *appartenant à* Φ_t, $i \neq i'$, *et tout site* k *et* k', $k \neq k'$ *tels que :*

$$c_{i'k} + c_{ik'} > c_{ik} + c_{i'k'}$$

aucune solution appartenant à $X_i^{k'} \cap X_{i'}^{k}$ *n'est optimale.*

Démonstration :

Soit une solution $x \in X_i^{k'} \cap X_{i'}^{k}$. Dans cette solution le client i est affecté au site k' et le client i' au site k.

Considérons la solution x' obtenue à partir de x en permutant l'affectation des clients i et i', de telle sorte que i soit affecté à k et i' à k'. L'affectation des autres clients n'est pas modifiée.

Pour tout site $j \neq k,k'$, l'affectation des clients n'a pas changée et il est donc évident que la configuration retenue pour ces sites dans la solution x reste optimale. Sachant que $i,i' \in \Phi_t$, cette permutation ne requiert pas de changement de configuration sur les sites k et k'. On a donc :

$$E_j(x') = E_j(x) \qquad j = 1,...,K$$

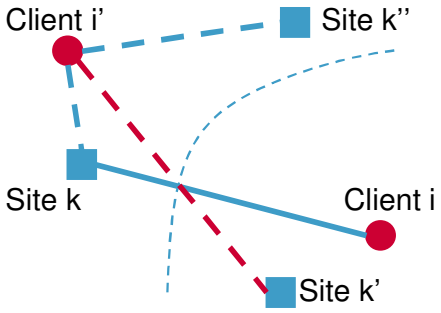


Figure 69: Illustration de la génération de coupes sur les distances

et part conséquent :

$$\sum_j E_j(x) = \sum_j E_j(x')$$

D'autres part :

$$\sum_{u=1}^{N}\sum_{j} x_{uj}\cdot c_{uj} = c_{i'k} + c_{ik'} + \sum_{u\neq i,i'}\sum_{j} x_{uj}\cdot c_{uj}$$
$$> c_{ik} + c_{i'k'} + \sum_{u\neq i,i'}\sum_{j} x'_{uj}\cdot c_{uj}$$
$$> \sum_{u}\sum_{j} x'_{uj}\cdot c_{uj}$$

On a donc :

$$\Gamma(x) > \Gamma(x') > \Gamma^{*}$$

La solution x n'est donc pas optimale.

L'application de ce résultat nous permet de réaliser des coupes lors de l'exploration de l'arbre du « Branch and Cut ». En effet, considérons une solution partielle obtenue en affectant les clients $1,...,N$ aux sites $a_1,...,a_N$. Considérons maintenant l'affectation du client $n+1$ au site a_{n+1}.

S'il existe $i \leq n$ tel que :

$$c_{n+1\,a_{n+1}} + c_{ia_i} > c_{ia_{n+1}} + c_{n+1\,a_i}$$

alors cette affectation ne peut conduire à une solution optimale. Pour voir cela, il suffit de voir que la solution x obtenue appartiendra, quelle que soit les affectations ultérieures des clients $n+2,...,N$, à $X_i^{k'} \cap X_{i'}^{k}$ avec $k' = a_i, i' = n+1$ et $k = a_{n+1}$. Cette solution ne peut être optimale d'après le résultat précédent.

H.9. Utilisation de la solution de l'heuristique searchCutExplore

Une autre technique pour accélérer la méthode exacte consiste à commencer l'exploration au voisinage de la solution fournie par l'heuristique searchCutExplore. Pour ce faire nous ordonnons les sites à visiter par ordre croissant de distance. Cette distance prend en compte la distance client site ajoutée à la distance entre ce site et le site le plus proche utilisée dans la solution de searchCutExplore. Cela à pour effet de démarrer l'exploration à partir d'une solution qui est déjà très proche de l'optimum.

I. Tests et résultats

L'un des premiers résultats que l'on tient à montrer est la pertinence du modèle que l'on a présenté section (D.). Sur une topologie test nous effectuons les tests suivants. Nous calculons la topologie optimale sans tenir compte des coûts des équipements et nous enregistrons le coût et les affectations des clients. Dans le deuxième test nous imposons l'affectation obtenue précédemment et nous calculons le coût de la topologie optimale en tenant compte cette fois des coûts des équipements.

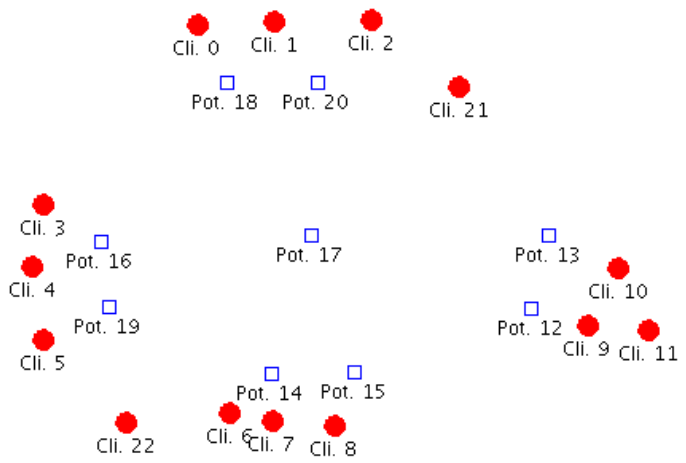


Figure 70: Topologie validation intérêt du modèle

Pour cette topologie nous avons 9 sites potentiels, 14 clients à raccorder qui génèrent 14 trafics.

Finalement nous calculons le coût de la topologie optimale sans imposer l'affectation précédente. Les résultats obtenus sont présentés dans le tableau suivant.

	Coût	écart
Solution sans coût d'équipement	4.21924 e6	
Solution avec coût d'équipement affectation fixée	6.02524 e6	+8.15 %
Solution optimale (avec coût d'équipement)	5.57089 e6	+0.00 %
Solution searchCutExplore (avec coût d'équipement)	5.67390 e6	+1.80 %

Tableau 11: Gain obtenu en tenant compte des coûts des équipements

Le fait de prendre en compte dès le départ du coût des équipements permet de faire une économie de l'ordre de 8,15% sur le coût de la topologie finale.

I.1. Topologies de tests

Dans ce qui suit nous allons présenter les topologies sur lesquelles nous effectuons nos tests. Nous disposons pour cela des équipements suivants :

Pour les routeurs nous utilisons les modèles suivants :

Modèle	# Slots	Capacité	Coût (K.€)
Modèle « Cisco1 »	12	384 (Mbps)	6
Modèle « Cisco2 »	16	1 (Gbps)	15
Modèle « Cisco3 »	8	10 (Gbps)	18

Tableau 12: Modèles de routeurs

Pour les interfaces nous utilisons les modèles suivants :

Modèle	# Port	Capacité (Mbps)	Coût (K.€)
Modèle « ITF16*2 »	16	16*2	2
Modèle « ITF12*10 »	12	12*10	3
Modèle « ITF8*100 »	8	8*100	5

Tableau 13: Modèles d'interfaces

Pour les liens nous utilisons les modèles suivants

Modèle	Coût kilométrique (€)
Modèle « 512kb »	25
Modèle « 2Mb »	55
Modèle « 10Mb »	80
Modèle « 100Mb »	115
Modèle « 1Gb »	145
Modèle « 10Gb »	165

Tableau 14: Modèles de liens

I.1.1. Topologie 1

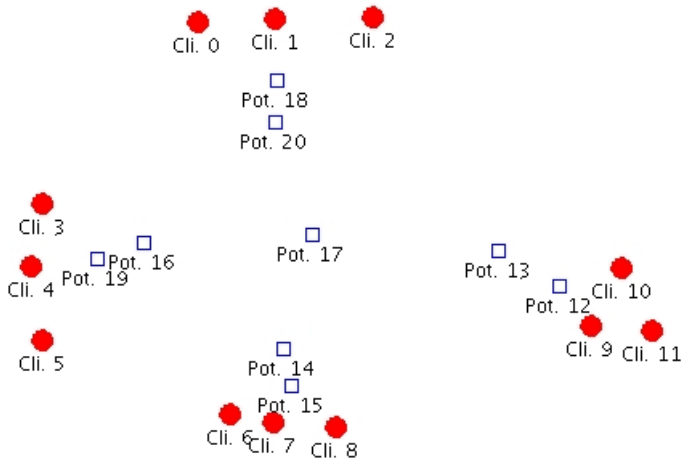


Figure 71: Topologie de test numéro 1

Pour cette topologie nous avons 9 sites potentiels, 12 clients à raccorder qui génèrent 12 trafics.

Nous effectuons 3 tests. Pour le premier test nous prenons des coûts de liens forts. Pour le deuxième des coûts de sites forts. Pour le troisième nous prenons les coûts réels sans facteurs multiplicatifs. Le but de ces trois tests consiste à montrer que de part la construction de l'heuristique quand la solution optimale est la solution centralisée ou la solution au plus prés l'heuristique la trouve forcement. Pour les autres cas l'heuristique fournit une assez bonne solution.

Test 1 (coût des liens prédominant) :

	Coût de la solution	Temps d'exécution	écart
SearchCutExplore	59668.85 (€)	0.06(s)	0.00%
Méthode exacte	59668.85 (€)	0.12(s)	

Tableau 15: Résultats topologie 1 avec coût des liens prédominant

Cette solution correspond à la solution au plus près, qui est testé par l'heuristique au cours de l'exploration du domaine des solutions. Il est tout à fait normal que l'on obtienne la solution optimale par l'heuristique.

Test 2 (coût des sites prédominant) :

	Coût de la solution	Temps d'exécution	écart
SearchCutExplore	515584.4 (€)	0.08	0.00%
Méthode exacte	515584.39 (€)	0.18	

Tableau 16: Résultats topologie 1 avec coût des sites prédominant

Cette solution correspond à la solution centralisée, celle ci est à la base de la construction de l'heuristique searchCutExplore. Il est donc normal que l'on trouve la solution optimale dans ce cas de figure aussi.

Test 3 (coûts réels) :

	Coût de la solution	Temps d'exécution	écart
SearchCutExplore	20584.4 (€)	1.07(s)	0.00%
Méthode exacte	20584.39 (€)	1.77(s)	

Tableau 17: Résultats topologie 1 avec coûts réels

Sur cette exemple l'heuristique fourni aussi la solution optimale nous verrons par la suite que ce n'est pas toujours le cas.

I.1.2. Topologie 2

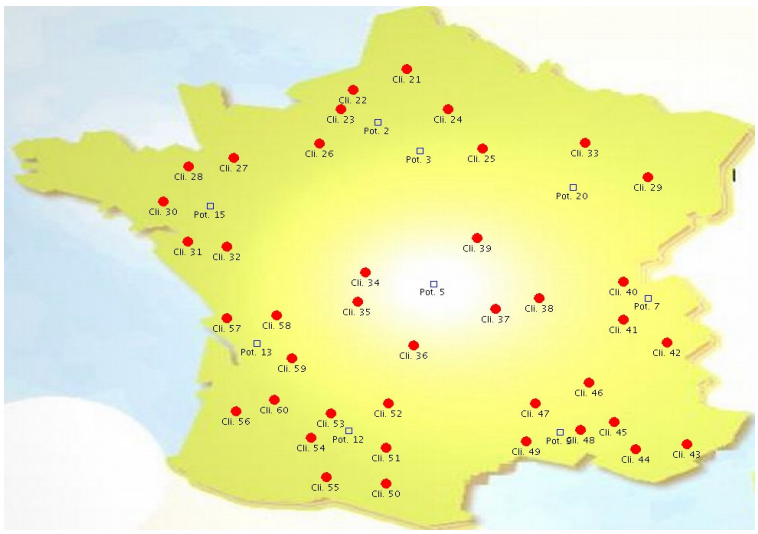


Figure 72: Topologie de test numéro 2

Pour cette topologie nous avons 9 sites potentiels, 40 clients à raccorder qui génèrent 400 trafics.

	Coût de la solution	Temps d'exécution	écart
SearchCutExplore	199395 (€)	0.27(s)	0.56%
Méthode exacte	198272 (€)	4469 (min)	
Méthode exacte++	198272 (€)	3 (min) 30 (s)	

Tableau 18: Résultats topologie 2

La solution obtenue est cette fois sous-optimale. La différence avec la solution exacte est de 0.56%. vu les temps d'exécutions nous pensons que celle reste de très bonne qualité.

I.1.3. Topologie 3

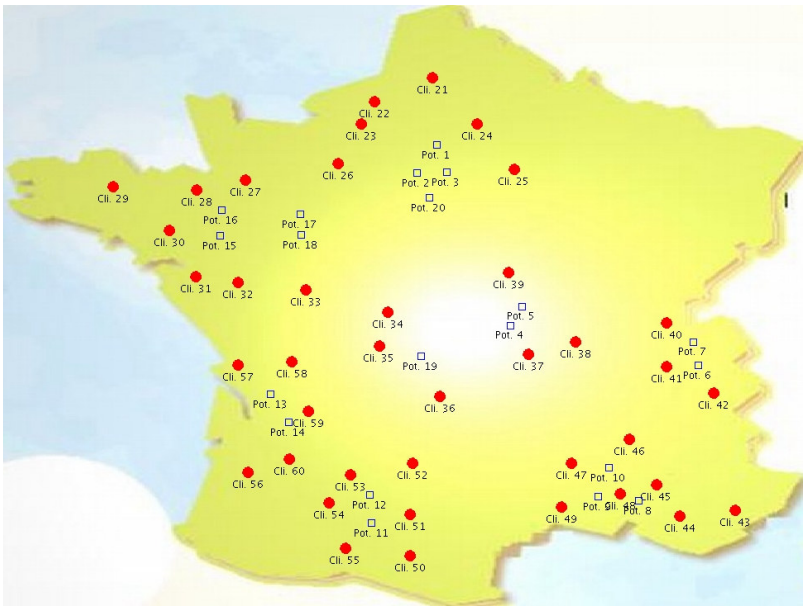


Figure 73: Topologie de test numéro 3

Pour cette topologie nous avons 20 sites potentiels, 40 clients à raccorder qui génèrent 400 trafics.

	Coût de la solution	Temps d'exécution	écart
SearchCutExplore	192946 (€)	0.77 (s)	0.29%
Méthode exacte	192381 (€)	11407 (min)	
Méthode exacte++	192381 (€)	168 (min)	

Tableau 19: Résultats topologie 3

La encore la solution obtenue par l'heuristique searchCutExplore est sous-optimale. L'écart avec la solution optimale est de 0.29%, il est tout à fait raisonnable de penser que encore une fois c'est une très bonne solution. Surtout lorsque l'on voit la différence entre les temps d'exécution.

I.1.4. Topologie 4

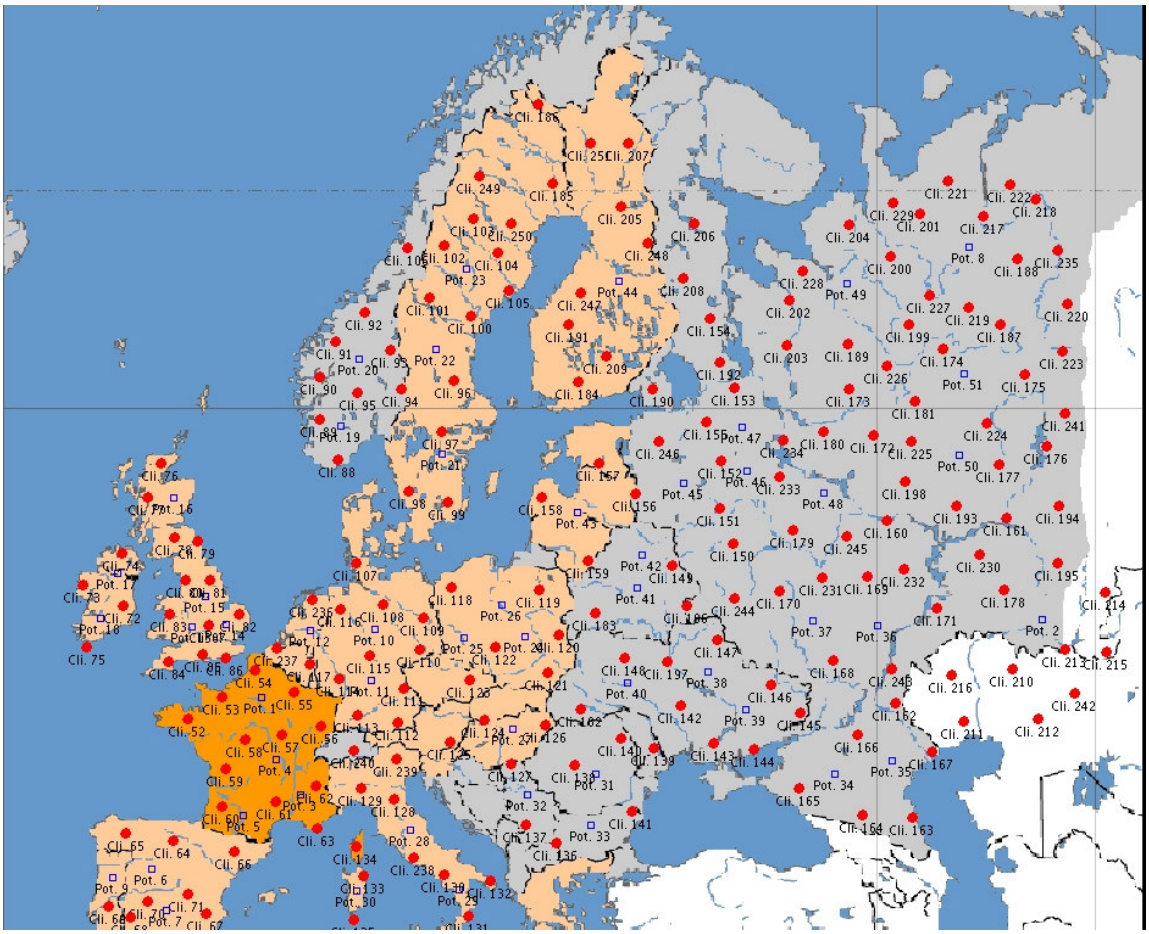


Figure 74: Topologie de test numéro 4

Pour cette topologie nous avons 40 sites potentiels, 200 clients à raccorder qui génèrent 400 trafics.

	Coût de la solution	Temps d'exécution	écart
SearchCutExplore	747637 (€)	1,49(s)	
Méthode exacte	----	----	

Tableau 20: Résultats topologie 4

Pour ce test la il est inenvisageable de tester la méthode exacte car il faudrait des mois de calcul. Compte tenu de la qualité des solutions obtenues sur les autres tests nous pensons que la solution obtenue par l'heuristique est une bonne solution.

J. Conclusion

Dans ce chapitre nous présentons une nouvelle méthodologie pour aborder le problème de la conception de réseau d'accès. L'idée principale est la prise en compte du trafic et des coûts de matériels. Cela permet de résoudre le problème de conception et de dimensionnement de la topologie. La double originalité réside dans la modélisation qui est une version plus fine du problème et dans les techniques de résolution proposées.

Nous proposons une modélisation originale (plus fine) du problème d'affectation traditionnellement étudié (affectation au niveau des cartes des routeurs). Celle-ci rend le problème plus complexe mais plus proche de la réalité des opérateurs. Nous présentons par la suite différents algorithmes pour la résolution de ce problème.

Nous proposons des algorithmes issu de la programmation linéaire basée sur une décomposition du problème initiale et sur le calcul des configurations optimales. Pour cette formulation nous présentons une méthode exacte (Branch and Bound amélioré) et un heuristique (relaxation Lagrangienne). La première permet d'obtenir la solution exacte et joue le rôle de référence. La deuxième vu la difficulté de convergence de l'algorithme dual permet au mieux de fournir une bonne borne supérieure. Celle ci peut être utilisée pour accélérer la méthode exacte.

Nous présentons aussi l'heuristique searchCutExplore qui fournit en des temps très rapides une solution proche de l'optimum (<1%) voire l'optimum dans certains cas. Celle ci est aussi utilisée comme « bonne » borne supérieure pour accélérer la méthode exacte.

CONCLUSION GENERALE

Les dernières évolutions technologiques poussent l'Internet d'aujourd'hui à prendre une place considérable dans la société actuelle. Parmi les acteurs de cette « révolution mondiale » on trouve les opérateurs qui ont poussé les limites de cet Internet jusqu'au plus proche des utilisateurs (UMTS, WIFI, WIMAX,…).

Ainsi la « *convergence* » des réseaux en un seul et unique réseau qui puisse servir tous types d'applications favorise l'émergence du protocole IP. Elle lui permet aussi de devenir de plus en plus l'élément fédérateur des réseaux de transport des données. En effet, la simplicité de sa gestion, de son déploiement et son caractère d'adaptabilité ainsi que le coût réduit de son infrastructure le rendent très facile à généraliser.

Avec la démocratisation du haut débit via l'ADSL ou autre UMTS, les opérateurs fournissent de plus en plus de services à forte valeur ajoutée. L'enjeu, guidé par une concurrence relevée, est d'augmenter les revenus de ces services tout en réduisant les coûts des infrastructures nécessaires à leur mise en place.

Bien qu'initialement prévu pour un service sans aucune garantie, l'Internet actuel achemine donc des paquets de toutes sortes d'applications (Voix, Vidéo,…). Cependant les marges des valeurs tolérées des délais, taux de pertes, gigues sont très variables en fonction de chaque application.

Il faut donc mettre en place une gestion de bout en bout de la QoS. En d'autres termes, cela signifie qu'il faut garantir les services qui génèrent des bénéfices tels que la voix ou la vidéo tout en attribuant une bande passante suffisante pour des applications moins critiques.

L'une des solutions proposées est l'association du protocole MPLS et du mécanisme DiffServ. Nous nous sommes basé sur cette association pour les travaux de cette thèse.

Ceux-ci, s'articulent autour de quelques idées simples.

- Tout d'abords une bonne gestion des ressources disponibles dans le réseaux est désormais un point crucial. En effet avec l'explosion des applications gourmandes en ressources nul opérateur ne peut négliger ce point en s'appuyant sur des politiques de surdimensionnement comme par le passé.

- Dans un second temps l'intégration de la prise en charge des indicateurs de QoS tel que le délai ou les pertes. C'est la condition ultime pour gérer les applications sensibles à la QoS telles que la Voix sur IP ou encore la vidéo.

- Dans un troisième temps la conception de topologie optimale permet de préparer le terrain. En effet concevoir une topologie en tenant compte du trafic ainsi que des contraintes matérielles permet de mieux gérer les ressources et de les disposer là ou on en a besoin.

Le besoin de méthodologie et d'outils de planification de réseaux est sans cesse grandissant pour concevoir et maintenir les réseaux actuels.

Ces outils se doivent dans la cas de la planification court-terme d'apporter, pour le routage des LSPs, des solutions garantissant une bonne gestion des ressources ainsi que le respect des contraintes de QoS, d'affinités et de résilience.

La conception de tels outils doit prendre en compte le facteur échelle au niveau du nombre de LSPs gérés, la taille des réseaux (qui peut être très grande), et que les opérateurs en ferons une utilisation régulière (aspect interactif).

Pour la planification long-terme ils doivent proposer des méthodes de conception de topologies optimales qui tiennent compte du trafic entre les utilisateurs ainsi que les contraintes des équipements.

Afin de répondre à ces questions, nous proposons plusieurs modélisations du problème de mono-routage des LSPs dans les réseaux MPLS. La première modélisation classique optimise le critère bande passante. La deuxième introduit la notion de QoS dans le critère d'optimisation par le biais d'un modélisation en files d'attente de type M/M/1/N. Et une troisième qui propose un modèle plus complexe de gestion de la QoS par la combinaison de files prioritaires et de files WFQ.

Pour résoudre les différents problèmes modélisés précédemment nous proposons une heuristique à trois niveaux ILSP-OLS-ACO qui exploite l'efficacité d'ILSP à générer les bons chemins « candidats », la rapidité et la précision de la solution multi-routée obtenue par OLS ainsi que la qualité de la solution mono-routée fournie par ACO.

Nous nous sommes intéressé par la suite au problème de sécurisation des LSPs. MPLS offre la possibilités d'avoir différents mécanismes de sécurisation (Backup, Fast Reroute, Multi Layer). Nous proposons différents algorithmes qui prennent en compte tous ces mécanismes.

Nous étudions par la suite le problème de conception de topologie d'accès. Nous proposons une modélisation qui tiens compte du trafic ainsi que du coût des équipements. Nous proposons dans ce cadre différentes approches. La première utilise la programmation linéaire et n'est utilisée que pour des problèmes de tailles relativement petites.

Nous proposons par la suite une amélioration de cette approche. En effet une version décomposée du problème basée sur le calcul des configurations optimales permet de traiter des problèmes de tailles plus importantes.

Nous proposons ensuite une heuristique qui combine une technique de « clustering » et une technique de recherche locale. Cette heuristique permet de traiter des topologies de tailles très grande en des temps plus que raisonnables. En outre les solutions obtenues sont à moins de 1% des solutions optimales obtenues par l'approche exacte.

TABLE DES ABBREVIATIONS

RFC	Request for Comment
IP	Internet Protocol
AS	Autonomous System
IGP	Internal Gateway Protocol
IGRP	Internal Gateway Routing Protocol
EGP	External Gateway Protocol
BGP	Border Gateway Protocol
e-BGP	External Border Gateway Protocol
i-BGP	Internal Border Gateway Protocol
OSPF	Open Shortest Path First
MPLS	Multi Protocol Label Switching
ATM	Asynchronous Transfer Mode
LER	Label Edge Router
LSR	Label Switch Router
FEC	Forwarding Equivalent Class
LSP	Label Switched Path
LSR	Label Switch Router
FT	Forward Table
CPE	Customer Premises Equipment
E-LSR	Edge Label Switch Router
FRR	Fast Re-Routing in MPLS
RSVP-TE	Resource Reservation Protocol Traffic Engineering
OSPF-TE	Open Shortest Path First Traffic Engineering
VPN	Virtual Private Network
FTN	FEC-To-NHLFE Map
ILM	Incoming Label Map
NHLFE	Next Hop Label Forwarding Entry
AF	Assured Forwarding
BA	Behaviour Aggregate
CS	Class Selector
DF	Default Forwarding
DSCP	Differentiated Services Code Point
EF	Expedited Forwarding
PHB	Per Hop Behaviour
OA	Ordered Aggregate
PSC	PHB Scheduling Class
E-LSP	EXP-Inferred-PSC LSP
L-LSP	Label-Only-Inferred-PSC LSP
QoS	Quality Of Service
IGP	Interior Gateway Protocol
EGP	Exterior Gateway Protocol
TOS	Type Of Service
SLA	Service Level Agreement
WFQ	Weighted Fair Queuing
WRR	Weighted Round Robin
PQ	Priority Queuing

GPS	Generalised Processor Sharing
CoS	Class Of Service
TDSA	Two Bit Differentiated Service Architecture
LSA	Link-State Advertisements
POS	Packet over Sonet/SDH

BIBLIOGRAPHIE

[ADA00] Adan I.J.B.F., Boxma O.J., Resing J.A.C. , (2000), Queueing models with multiple waiting lines. SPOR-reports.

[ALT92] Altinkemer K. and Yu Z., (1992), Topological design of wide area communication networks. *Annals of Operations Research*, 36:365–382.

[ALT93] Altinkemer K. and Chaturvedi A., (1993), Neural networks for topological design of local access tree networks. In *Proceeding of Telecommunication Systems, Modelling and Analysis Conference*, pages 256–263.

[AND98] Andrews M. and Zhang L., (1998), The Access Network Design Problem. In *39th Annual Symposium on Foundations of Computer Science*, pages 40–49.

[ASH98] G. R. Ash, (1998), *Dynamic Routing in Telecommunications Networks,* McGraw Hill.

[BAL94] Balakrishnan A., Magnanti T.L., and Mirchandi P., (1994), Modeling and heuristic worst-case performance analysis of the two-level network design problem. *Management Science*, 40(7):846–867.

[BAN97] Bandhari R., Optimal Physical Diversity Algorithms and Survivable Networks, 1997 IEEE.

[BEN96] Bennett J. C. R., Zhang H. (1996) WF2Q : Worst-case Fair Weigthed Fair Queuing. In Procesdings of INFOCOM, pp 120-128.

[BER04] Berger R.T. and Raghavan S., (2004), Long-Distance Access Network Design. *Management Science*, 50:309–325, 2004.

[BER92] Bertsekas D. and Gallager R. (1992). Data Networks, second edition. Prentice-Hall.

[BOC05] Bockstal C. (2005) Modélisation Différentielle du trafic et Simulation hybride de réseaux IP-MPLS DiffServ, Thèse LAAS-CNRS, pp 49-60.

[BOG93] Bogdanowicz Z.R., (1993), A new optimal algorithm for backbone topology design in communications networks. *Mathematical and Computer Modelling*, 17(8):49–61.

[BON99] E. Bonabeau, M. Dorigo and G. Theraulaz, (1999), *Swarm Intelligence - From Natural to Artificial Systems,* Oxford University Press.

[BOO77] Boorstlyn R.R. and Frank H., (1977), Large-scale network topological optimization. *IEEE Transactions on Communications*, 25(1):29–47.

[BOX03] Boxma O. J., Borst S.C., Jelenkovic P (2003). Reduced-load equivalence and induced burstiness in GPS queues with long-tailed traffic flows. Queueing Systems 43, pp 273-306.

[BRO70] Broyden C. G. (1970), The convergence of a class of double-rank minimization algorithms 2: the new algorithm, Journal Institute of Math. And its Appl. 12, pp 223-245.

[BRU02] Brun O., Garcia J.M., Bockstal C., (2002), Hyperexponential approximation for GPS systems with an arbitrary number of traffic classes, Rapport LAAS N°02003, 21p.

[CHA01] Chamberland S., Sanso B., (2001), On the design problem of multitechnology networks. *INFORMS Journal on Computing*, 13(3):245–256.

[CHA89] Chattopadhyay N.G., Morgan T.W., and Ranghuram A., (1989), An innovative technique for backbone network design. *IEEE Transactions on Systems, Man and Cybernetics*, 19(5):1122–1132.

[CHA98] P. Chanas, (1998), *Réseaux ATM: conception et optimisation,* PhD thesis, France Telecom CNET, Sophia-Antipolis, France.

[CHA99] P. Chanas, O. Goldschmidt and M. Burlet, (1999), *Routing Virtual Paths in ATM Networks,* Journal of Heuristics.

[CHE01] Cheriyan J., Sebo A., and Szigeti Z., (2001), Improving on the 1.5-approximation of a smallest 2-edge connected spanning subgraph. *SIAM Journal on Discrete Mathematics*, 14(2):170–180.

[COR99] D. Corne, M. Dorigo and F. Glover Eds, (1999), *New Ideas in Optimization,* McGraw Hill.

[COS94] S. Cosares and I. Saniee, (1994), *An optimization problem related to balancing loads on SONET rings,* Telecommunication Systems, 3:165-181.

[CRU99] Cruz F.R.B., MacGregor Smith J., Mateus G.R., (1999), Algorithms for a multi-level network optimisation problem. *European Journal of Operational Research*, 118:164–180.

[CUI03] CUI J., KIM J., MAGGIORINI D., BOUSSETTA K., GERLA M., (2003), « Aggregated Multicast: A Comparative Study », Special Issue of Cluster Computing: The Journal of Networks, Software and Applications.

[DAV93] Davis L., Orvosh D., Cox A., and Qiu Y., (1993), A genetic algorithm for survivable network design. In *Proceedings of the Fifth International Conference on Genetic Algorithms (San Mateo, CA, USA)*, pages 408–415.

[DEN83] Deneubourg J.-L., Pasteels J.-M. et Verhaeghe J.-C., (1983). « Probabilistic Behaviour in Ants: a Strategy of Errors ? », *Journal of Theoretical Biology*, 105.

[DOR00] Dorigo M., Bonabeau E., Theraulaz G., (2000), «Ant algorithms and stimergy», Future Generation Computer Systems, vol. 16, p. 851-871.

[DUH03] Duhamel C., Vatinlen B., Mahey P., Chauvet F., (2007), Minimizing congestion with a bounded number of paths, à paraître dans EJOR(2007), 6p.

[ESQ02] Garcia J.M. et al, (2002), *Modélisation des mécanismes DiffServ dans les cœurs de réseaux MPLS* (Projet RNRT ESQUIMAUX), Rapport de recherche LAAS-CNRS.

[EST98] ESTRIN D., FARINACCI D., HELMY A., THALER D., DEERING S., HANDLEY M., JACOBSON V., LIU C., SHARMA P., WEI L., (1998), « Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification », IETF RFC 2362.

[FAR00] FARINACCI D., REKHTER Y., ROSEN E., (2000), « Using PIM to distribute MPLS labels for multicast routes », IETF Internet draft.

[FER98] Fernandes C.G., (1998), A better approximation ratio for the minimum k-edge-connected spanning subgraph problem. *Journal of Algorithms*, 28(1):105–124.

[FLE64] Fletcher R., Reeves C. M (1964), Function minimization by conjugate gradients, computer Journal 7, pp. 149-154.

[FLE70] Fletcher R (1970), A new approach to variable metric algorithms, The Computer Journal, vol. 13, n°3,pp.317-322.

[FRA73] Fratta L., Gerla M., and Kleinrock L. (1973). The flow deviation method: An approach to store-and-forward communication network design. Networks, 3:97–133.

[FRA99] Frantzeskakis L.F. and Luss H., (1999), The network redesign problem for access telecommunications networks. *Naval Research Logistics*, 46:487–506.

[FRE99] C. Frei and B. Faltings, (1999), *Bandwidth allocation heuristics in communication networks,* 1ère Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (ALGOTEL'99), 53-58, Roscoff, France.

[GAL04] Galluccio A. and Proietti G., (2004), Polynomial time algorithms for edge-connectivity augmentation problems. *Algorithmica*, 36(4):361–374.

[GAV86a] Gavish B., (1986), A general model for the topological design of computer networks. In *GLOBECOM'86 - IEEE International Global Telecommunications Conference*, pages 1584–1588.

[GAV86a] Gavish B. and Altinkemer K., (1986), Parallel savings heuristics for the topological design of local access tree networks. In *Proceedings of IEEE INFOCOM'86. Fifth Annual Conference on Computers and Communications Integration Design, Analysis, Management*, pages 130–139 .

[GAV91] Gavish B., (1991), Topological design of telecommunication networks - local access design methods. *Annals of Operations Research*, 33:17–71.

[GAV92a] Gavish B., (1992), Configuring wide area computer networks-problems and models. *OR Spektrum*,14(3):115–128.

[GAV92b] Gavish B., (1992) Topological design of computer communication networks-the overall design problem. *European Journal of Operations Research*, 58(2):149–172.

[GHO02] Ghosh L., Mukherjee A., and Saha D., (2002), Design of 1-ft communication network under budget constraint. In *Proceedings of Distributed Computing. Mobile and Wireless Computing. 4th International Workshop - IWDC 2002 (Calcutta, India)*, pages 300–311, 2002.

[GIR01] Girard A., Sanso B., and Dadjo L., (2001), A tabu search algorithm for access network design. *Annals of Operations Research*, 106(1-4):229–262.

[GUE99] Guérim R., Peris V.(1999) Quality-of-Service in Packet Networks Basic Mechanisms and Directions. Computer Network, No.31,pp.169-189.

[GOL70] Goldfarb D. (1970), A Family of variable metric methods derived by variational means, Mathematics of Computation, 24, pp. 23-26.

[GOL94] Golestani R. J. (1994) A self-clocked fair queueing scheme for broadband applications. In Porceeding of INFOCOM, pp 636-646.

[GOU97] Gouveia L. and Lopes M.J., (1997), Using generalized capacitated trees for designing the topology of local access networks. *Telecommunications Systems*, 7(4):315–337.

[HAK64] Hakimi, S.L., (1964), Optimum Locations of Switching Centers And The Absolute Centers and Medians of a Graph, Operations Research, (12), 450-459.

[HAK65] Hakimi, S.L., (1965), Optimum Distribution of Switching Centers in a communication Network and Some Related Graph Theoretic Problems, Operations Research, (13), 462-475.

[JAE92] Jae gyun Kim and Dong wan Tcha, (1992), Optimal design of a two-level hierarchical network with tree-star configuration. *Computers & Industrial Engineering*, 22(3):273–281.

[JAI01] Jain K., (2001), A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21:39–60.

[KAN02] Kanungo T., and al., (2002), An Efficient k-Means Clustering Algorithm: Analysis and Implementation, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 24 , Issue 7, Pages: 881 – 892.

[KIR83] Kirkpatrick S., Gelatt C.D. and Vecchi M. P., (1983), *Optimization by simulated annealing,* Science 220(4598):671-680.

[KLE76] Kleinrock L., Queueing Systems, (1976), « Volume II: Computer Applications » A Wiley-Interscience Publication.

[KHU91] Khuller S. and Thurimella R., (1991), Approximation algorithms for graph augmentation. *Journal of Algorithms*, 4(2):214–225.

[KON99] Konak A. and Smith A.E., (1999), A hybrid genetic algorithm approach for backbone design of communication networks. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Washington, DC, USA)*, volume 3, pages 1817–1823.

[KOS02] Kos M., Mikac M., and Mikac D., (2002), Topological planning of communication networks. *Journal of Information and Organizational Sciences*, 26(1-2):57–68.

[KRY01] Krysta P. and Kumar V.S.A., (2001), Approximation algorithms for minimum size 2-connectivity problems. In *Proceedings of the 18th Annual Symposium Theoretical Aspects of Computer Science (STACS'01,Berlin), Lecture Notes in Computer Science*, volume 2010, pages 431–442. Springer.

[LUE73] Luenberger D. G., (1973), Introduction to linear and nonlinear programming, addison Wesley.

[LUK99] Lukic B., (1999), An approach of designing local access network using Simulated Annealing method. In *ConTEL'99 - 5th International Conference on Telecommunications*, pages 241–247.

[MAG94] Balakrishnan A., Magnanti T.L., and Mirchandi P., (1994), A dual-based algorithm for multi-level network design. *Management Science*, 40:567–581.

[MAH01] Mahey P., Randazzo C.D., Luna H.P.L., (2001), Benders decomposition for local access network design with two technologies. *Discrete Mathematics & Theoretical Computer Science*, 4:235–246.

[MAH06] Mahey P., Duhamel C., (2006) Multicommodity Flows with a bounded number of paths – A Flow Deviation Approach, Networks, Vol. 49, n°1,pp 80-89.

[MAN03] Mandal S., Saha D., Mukherjee R., and Roy A., (2003), An efficient algorithm for designing optimal backbone topology for a communication networks. In *Proceedings of ICCT 2003 - International Conference on Communication Technology (Beijing, China)*, volume 1, pages 103–106.

[MAT00] Mateus G.R. and Franqueira R.V.L., (2000), Model and heuristics for a generalized access network design problem. *Telecommunication Systems - Modelling, Analysis, Design and Management*, 15(3-4):257–271.

[MIN74] M. Minoux, (1974), *Planification à court et à moyen terme d'un réseau de télécommunications,* Annales des Télécommunications, Vol. 29, No 11-12.

[MIN83] Minoux M., (1983), Programmation Mathématique, Dunod ed: Tome 1.

[MIR98] Balakrishnan A., Magnanti T.L., and Mirchandi, (1998), Designing hierarchical survivable networks. *Operations Research*, 46(1):116–136.

[MOY88] Moyson F. et Manderick B., (1988). « The Collective Behaviour of Ants: an Example of Self-Organisation in Massive Parallelism », *Proceedings of the AAAI Spring Symposium on Parallel Models of Intelligence.* Stanford, California.

[MUK02] Ghosh L., Mukherjee A., and Saha D., (2002), Optimal design of backbone topology for a communication network cost constraint. In *Proceedings of ICCC 2002 - 15th International Conference on Computer Communication*, volume 2, pages 471–485, 2002.

[NUT97] Nutov Z. and Penn M., (1997), Faster approximation algorithms for weighted triconnectivity augmentation problems. *Operations Research Letters*, 21:219–223.

[OUO00] Ouorou A., Mahey P., Vial V.P., (2000), A survey of algorithms for convex multicommodity Flow Problems, Management Science, Vol. 46, n° 1, pp. 126-147.

[PEN97] Michael Penn and Haya Shasha-Krupnik, (1997), Improved approximation algorithms for weighted 2- and 3- vertex connectivity augmentation problems. *Journal of Algorithms*, 22:187–196.

[PIE97] Pierre S. and Elgibaoui A., (1997), A tabu-search approach for designing computer-network topologies with unreliable components. *IEEE Transactions on Reliability*, 46(3):350–359.

[PRI99] Priem M. and Priem F., (1999) *Ingénierie des WAN (text in French)*. Dunod InterEditions.

[PRO99] Provan J.S. and Burk R.C., (1999), Two-connected augmentation problems in planar graphs. *Journal of Algorithms*, 32:87–107.

[RAC00] Rachdi M.A., (2000), *Routage Optimal des VP/VC dans les réseaux ATM*, Rapport de Stage ISIMA.

[RAC03] Rachdi M.A., Garcia J.M., Brun O., (2003), *Optimal LSP Placement with QoS constraints in IP/ DiffServ /MPLS networks*. ITC18 issue.

[RAN01] Randazzo C.D. and Luna H.P.L., (2001), A comparison of optimal methods for local access uncapacited network design. *Annals of Operations Research*, 106:263–286.

[ROM03] Rombaut M. 2003 , Optimisation des réseaux, routage et dimensionnement. doctorat de l'Université de Versailles Saint-Quentin-en Yvelines.

[ROS60] Rosen J. B. (1961), The gradient projection method for non-linear programming, part1: linear constraints, Journal S.I.A.M., 8, p. 181-217.

[SAN00] Sanso B., Chamberland S., Marcotte O., (2000), Topological design of two-level telecommunication networks with modular switches. *Operations Research*, 48(5):745–760.

[SCH97] R. Schoonderwoerd, O. E. Holland, J. L. Brutten and L. J. M. RothKrantz, (1997), *Ant based load balancing in telecommunication networks,* Adaptative Behaviour, 5(2).

[SHA69] Shano D. F. (1970), Conditioning of quasi-newton methods for function minimization, Mathematics of Computation,24, pp. 641-656.

[SUN92] Sung hark Chung, Young soo Myung, and Dong wan Tcha, (1992), Optimal design of a distributed network with two-level hierarchical structure. *European Journal of Operational Research*, 62(1):105–115.

[TAN00] Tang Z., and Chiu A., (2000), Performance Analysis on Weigthed Fair Queueing Fifth INFORMS Telecommunications Conference, Boca Raton, Florida.

[THO02] Thomadsen T. and Clausen J., (2002), Hierarchical network design using simulated annealing. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 305, DK-2800 Kgs. Lyngby, sep 2002.

[TRA98] Tran L. and Beling P.A., (1998), A heuristic for the topological design of two-tiered networks. In *SMC'98 - IEEE International Conference on Systems, Man, and Cybernetics*, volume 3, pages 2962–2967.

[TZO98] Sheng-Tzong Cheng, (1998), Topological optimization of a reliable communication network. *IEEE Transactions on Reliability*, 47(3):225–233.

[VEM00] Vempala S. and Vetta A., (2000), Factor 4/3 approximations for minimum 2-connected subgraphs. In *Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (Berlin, Germany)*, pages 262–273.

[WAT93] Watanabe T. and Nakamura A., (1993), A minimum 3-connectivity augmentation of a graph. *Journal of Computer and System Sciences*, 46(1):91–128.

[ZHI01] Zhi-Zhong Chen, (2001), Approximating unweighted connectivity problems in parallel. *Information and Computation*, 171:125–126.

[RFC791] : RFC IP
http://www.ietf.org/rfc/rfc0791.txt

[RFC3031] : RFC MPLS
http://www.ietf.org/rfc/rfc3031.txt?number=3031

[RFC3032] : RFC MPLS
http://www.ietf.org/rfc/rfc3032.txt?number=3032

Forum MPLS
http://www.mfaforum.org/

[RFC3812] : RFC MPLS-TE
http://www.ietf.org/rfc/rfc3812.txt

[RFC3209] : RFC RSVP-TE
http://www.ietf.org/rfc/rfc3209.txt?number=3209

[RFC4090] : RFC Fast Reroute
http://www.ietf.org/rfc/rfc4090.txt?number=4090

[RFC4561] : RFC Fast Reroute
http://tools.ietf.org/html/rfc4561

[RFC2475] : RFC Diffserv
http://www.ietf.org/rfc/rfc2475.txt

[RFC1633] : RFC Intserv
http://www.ietf.org/rfc/rfc1633.txt?number=1633

[RFC3353] : RFC Multicast
http://www.ietf.org/rfc/rfc3353.txt

[DRAFT05] : Draft MPLS Multicast
http://bgp.potaroo.net/ietf/all-ids/draft-boudani-mpls-multicast-tree-05.txt

[RFC4205] : RFC SRLG
http://tools.ietf.org/html/rfc4205

[RFC4206] : RFC SRLG
http://tools.ietf.org/html/rfc4206

[Cisco LLQ] Cisco LLQ
http://www.cisco.com/warp/public/788/voice-QoS/voip-mlppp.html

[RFC2205] RFC RSVP
http://www.ietf.org/rfc/rfc2205.txt

[RFC3988] RFC CR-LDP
http://www.ietf.org/rfc/rfc3988.txt?number=3988

[RFC2328] RFC OSPF
http://www.ietf.org/rfc/rfc2328.txt?number=2328

[RFC1771] RFC BGP
http://www.ietf.org/rfc/rfc1771.txt